Control-Oriented Model Reduction for Parabolic Systems

vorgelegt von

Diplom-Mathematikerin
Ulrike Baur
aus Berlin

Von der Fakultät II - Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktorin der Naturwissenschaften
- Dr. rer. nat. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. John M. Sullivan
Berichter:	Prof. Dr. Peter Benner
Berichter:	Prof. Dr. Volker Mehrmann
Gutachter:	Prof. Dr. Dr. h.c. Wolfgang Hackbusch

Tag der wissenschaftlichen Aussprache: 23.01.2008

Berlin 2008
D 83

Zusammenfassung

In der vorliegenden Arbeit werden effiziente Implementierungen von Modellreduktionsverfahren für lineare, zeitinvariante Kontrollsysteme entwickelt. Modellreduktion spielt eine wichtige Rolle in der Simulation, Kontrolle und Optimierung von komplexen, dynamischen Systemen, wie sie bei der Modellierung vieler physikalischer Prozesse und bei der Ortsdiskretisierung parabolischer Differentialgleichungen auftreten. Wird die Finite-Elemente-Methode (FEM) zur Ortsdiskretisierung einer partiellen Differentialgleichung benutzt, besteht das resultierende System gewöhnlicher Differentialgleichungen typischerweise aus vielen Gleichungen, die dazugehörigen Matrizen sind schwachbesetzt. Wird allerdings die Randelementmethode (BEM) angewendet, sind die entstehenden Systeme groß und vollbesetzt. Diese Problemklasse kann in einem sogenannten „quasi-dünnbesetzten“ Matrixformat, den hierarchischen ($\mathcal{H}$) Matrizen, approximiert werden [89]. Um die Problemgröße von linearen, zeitinvarianten Systemen zu reduzieren, wird sehr häufig das Verfahren des balancierten Abschneidens eingesetzt. Bei dieser Methode führt ein systemtheoretischer Hintergrund zu günstigen Eigenschaften im reduzierten System, so bleibt die Stabilität des Systems erhalten, und es existiert eine globale, berechenbare Fehlerschranke. Ein wesentlicher Nachteil der Methode ist der große Rechenaufwand, verursacht durch das Lösen zweier Lyapunovgleichungen bei kontinuierlichen Systemen und zweier Steingleichungen bei diskreten Systemen. In den letzten Jahren wurden viele effiziente Verfahren zur Lösung dieser Matrixgleichungen entwickelt. Die meisten sind besonders für schwachbesetzte Matrizen geeignet und berechnen Niedrigrangfaktoren der Lösungen. Allerdings ist der Rechenaufwand der Methoden kubisch, wenn sie auf große, vollbesetzte Systeme angewandt werden, und der Speicherbedarf wächst quadratisch.

In dieser Arbeit wird der Rechenaufwand und der Speicherbedarf der Algorithmen zur Lösung von Matrixgleichungen für große, quasi-dünnbesetzte Systeme reduziert. Für diese, in vielen Anwendungen auftretende Problemklasse, werden effiziente Algorithmen zur Lösung von Sylvester-, Lyapunov- und algebraischen Bernoulligleichungen auf Basis der Matrix-Signumfunktionsmethode, zur Lösung von Steingleichungen mit der quadrierten Smith-Iteration, entwickelt. Durch das Ersetzen der Matrixinversion, der Addition und der Multiplikation durch approximative Arithmetik für hierarchische Matrizen erhalten wir Algorithmen mit linear-polylogarithmischer Komplexität. Die Verfahren sind somit auf Systeme großer Ordnung anwendbar, in denen die Koeffizientenmatrix vollbesetzt sein darf, solange sie als $\mathcal{H}$-Matrix approximiert werden kann. Alle Methoden berechnen approximative Niedrigrangfaktoren der Lösungen der Matrixgleichungen.

Aufbauend auf den entwickelten Methoden zur Lösung großer Matrixgleichungen werden effiziente Implementierungen verschiedener Modellreduktionsverfahren vorgeschlagen. Neben dem approximativen balancierten Abschneiden, welches gute Approximationseigenschaften für große Frequenzen besitzt, wird die singuläre Störungsapproximationsmethode (SPA) modifiziert, die den stationären Zustand des Systems gut approximiert. Ein modifizierter Cross-Gramian Ansatz berechnet auf Basis der Lösung einer Sylvestergleichung redu-

zierte Systeme für symmetrische Systeme und für Systeme mit skalarem Ein- und Ausgang. Der Fehler ist vergleichbar klein zum approximativen balancierten Abschneiden. Weiterhin wird ein Ansatz zur Modellreduktion von instabilen Systemen behandelt, der auf einem Löser für algebraische Bernoulligleichungen und dem Löser für Lyapunovgleichungen basiert. Die Methoden werden erfolgreich auf Systeme resultierend aus FEM- und BEM-Diskretisierungen von zwei- und dreidimensionalen partiellen Differentialgleichungen der Größenordnung $\mathcal{O}(10^5)$ angewendet. Beachtenswert hierbei ist, dass die Anwendung des balancierten Abschneidens auf vollbesetzte Systeme dieser Größenordnung nur durch den Gebrauch des speziellen quasi-dünnbesetzten Matrixformates und der dazugehörigen approximativen Arithmetik möglich wird. Theoretisch und anhand der numerischen Beispiele wird die Beschränktheit des durch die $\mathcal{H}$-Matrix Approximation eingeführten Fehlers zwischen Originalsystem und dem System reduzierter Ordnung gezeigt. Alle Methoden reduzieren die Dimension des Systems beträchtlich und halten dabei eine kleine Fehlertoleranz ein. Eine stark reduzierte Problemgröße kann vorteilhaft in linear-quadratischen Optimalsteuerungsproblemen mit Ungleichheitsnebenbedingungen an die Steuerung ausgenutzt werden. Die üblicherweise sehr große Dimension des analogen diskreten quadratischen Problems wird durch Reduktion der Dimension der zugrundeliegenden PDE signifikant reduziert. Dadurch kann Standardsoftware zur Lösung des Optimalsteuerungsproblems eingesetzt werden.

Abstract

In the present work, the efficient implementation of model order reduction methods for linear, time-invariant control systems is investigated. Model reduction is common in simulation, control and optimization of complex dynamical systems arising in modeling of physical processes and in the spatial discretization of parabolic partial differential equations (PDEs) in three or more dimensions. If the finite element method (FEM) is used for the spatial discretization of PDEs, then the resulting system matrices are typically large and sparse; if the boundary element method (BEM) is applied, then the matrices are fully populated but often allow for a data-sparse representation. A suitable data-sparse matrix format for this problem class is provided by the hierarchical ($\mathcal{H}$) matrix format [89]. In systems theory and control of ordinary or partial differential equations, balanced truncation and related methods are very popular in model order reduction since they have some desirable properties: they preserve the stability of the system and provide a global computable error bound. The major drawback of balanced truncation is the high computational complexity caused by the solution of two Lyapunov equations for continuous-time systems and of two Stein equations for discrete-time systems. Several approaches to the solution of large-scale matrix equations were derived in the last two decades. Some of them are suitable for large-scale, sparse systems and especially adapted for the purpose of model order reduction as they exploit the typical low-rank property of the solution by computing approximate low-rank solution factors. However, these methods are of cubic complexity when applied to dense problems.

In this work, the complexity and the storage requirements of several solvers for matrix equations are reduced for the practically relevant class of data-sparse systems. For the numerical solution of Sylvester, Lyapunov and algebraic Bernoulli equations the sign function method is employed; for Stein equations the squared Smith iteration is considered. Replacing the usual matrix inversion, addition and multiplication by formatted arithmetic for hierarchical matrices, implementations with linear-polylogarithmic complexity and memory requirements are obtained. Efficient implementations of balancing-related model reduction methods based on these data-sparse solvers are developed. Besides an approximate balanced truncation method, an implementation of singular perturbation approximation is described and a method based on approximate low-rank factors of the cross-Gramian is proposed. Moreover, an approach for model order reduction of unstable systems based on the data-sparse solution of algebraic Bernoulli equations and of Lyapunov equations is derived. The methods are successfully applied on systems of order $\mathcal{O}(10^5)$ coming from FEM and BEM discretizations of two- and three-dimensional parabolic PDEs also including varying diffusion coefficients or convective terms. Note that the application of balanced truncation on dense systems of this size is only feasible if the data-sparse format and the corresponding formatted arithmetic is invoked. It is shown, both theoretically and in the numerical experiments, that the error between the original and the reduced-order system introduced by using the $\mathcal{H}$-matrix format is bounded. All methods significantly reduce the dimension of the

systems, while satisfying a very small error tolerance. A very small dimension can be exploited in linear-quadratic optimal control problems where inequality constraints for the control are given. The typically very large dimension of the corresponding discrete quadratic programming problem is significantly reduced if the underlying PDE dimension is reduced by the approximate balanced truncation method. Thus, standard optimization software can be applied for the solution of the optimal control problem.

Acknowledgment

I thank especially

- Peter Benner for his support, his interest in this work, his friendly manner and his patience;
- Daniel Kressner for many helpful suggestions, constructive remarks and careful proof reading;
- Lars Grasedyck for stimulating discussions and for many helpful comments regarding the hierarchical matrices;
- the system administrators, especially Jörn Beutner, for the excellent technical support;
- Uwe Prüfert for his suggestions on control-constrained optimal control problems;
- Achim Ilchmann for his encouragement and constant support;
- and, finally, my parents who made this all possible in the first place.

Contents

Abbreviations

ABE	algebraic Bernoulli equation
ADI	alternating direction implicit
ARE	algebraic Riccati equation
BEM	boundary element method
BT	balanced truncation
CG	cross-Gramian
DAE	differential algebraic equation
EPS	machine precision
FEM	finite element method
flops	floating-point arithmetic operations
GB	gigabyte
$\mathcal{H}$	hierarchical
HSV	Hankel singular value
LMI	linear matrix inequality
LQG	linear-quadratic-Gaussian
LTI system	linear, time-invariant system
MB	megabyte
MEMS	micro-electro-mechanical systems
MIMO system	multi-input, multi-output system
ODE	ordinary differential equation
PDE	partial differential equation
RLC	resistor-inductor-capacitor
RRLQ	rank-revealing LQ factorization
RRQR	rank-revealing QR factorization
SISO system	single-input, single-output system
SPA	singular perturbation approximation
SVD	singular value decomposition
TFM	transfer function matrix

Symbols

$\jmath$	$=$	$\sqrt{-1}$
$\mathrm{Re}\,(z)$	$=$	a, real part of a complex number $z = a + \jmath b,\ a,\ b \in \mathbb{R}$
$\mathrm{Im}\,(z)$	$=$	b, imaginary part of a complex number $z = a + \jmath b\ a,\ b \in \mathbb{R}$
$\jmath\mathbb{R}$		imaginary axis
ln		natural logarithm
$\log_2$		logarithm to basis 2
$\log_{10}$		logarithm to basis 10
$\#M$		cardinality of a set M
$\mathcal{N}_{op}(x)$		number of flops for a certain operation/method with dependency on parameters x
$\dot{\bigcup}$		disjoint union
$\square$		end of example, proof, remark

Sets

$\mathbb{N}$		set of natural numbers (including 0)
$\mathbb{Z}$		set of integers
$\mathbb{R}$		set of real numbers
$\mathbb{C}$		set of complex numbers
$\mathbb{C}^-$	$=$	$\{s \in \mathbb{C} \,\vert\, \mathrm{Re}\,(s) < 0\}$
$\mathbb{C}^+$	$=$	$\{s \in \mathbb{C} \,\vert\, \mathrm{Re}\,(s) > 0\}$
$\overline{\mathbb{C}^+}$	$=$	$\{s \in \mathbb{C} \,\vert\, \mathrm{Re}\,(s) \geq 0\}$
$\mathbb{C}_\alpha$	$=$	$\{s \in \mathbb{C} \,\vert\, \mathrm{Re}\,(s) > \alpha,\, \alpha \in \mathbb{R}\}$
$\mathcal{D}$	$=$	$\{z \in \mathbb{C} \,\vert\, \vert z\vert < 1\}$, the unit disc
$\mathcal{D}^+$	$=$	$\{z \in \mathbb{C} \,\vert\, \vert z\vert > 1\}$, the complement of the closed unit disc
$\mathcal{D}_\gamma$	$=$	$\{z \in \mathbb{C} \,\vert\, \vert z\vert > \gamma\}$ for $\gamma > 0$

Matrices

$\Lambda(A)$		spectrum of a matrix $A \in \mathbb{C}^{n\times n}$
$\rho(A)$	$=$	$\max\{\vert\lambda\vert : \lambda \in \Lambda(A)\}$, spectral radius of $A \in \mathbb{C}^{n\times n}$
$\sigma_{\max}(A)$		largest singular value of a matrix $A \in \mathbb{C}^{n\times m}$
$\mathrm{cond}_2\,(A)$	$=$	$\Vert A\Vert_2\Vert A^{-1}\Vert_2$, 2-norm condition number of $A \in \mathbb{C}^{n\times n}$
$n_\tau(A)$		numerical rank of a matrix $A \in \mathbb{C}^{n\times m}$ w.r.t. threshold $\tau \geq 0$
A^H	$=$	$(\bar{A})^T$, complex conjugate transpose of $A \in \mathbb{C}^{n\times m}$
e_i	$=$	$[\underbrace{0,\ldots,0}_{i-1},1,\underbrace{0,\ldots,0}_{n-i}]^T$, ith unit vector in $\mathbb{R}^n$
I_n	$=$	$[e_1,\ldots,e_n]$, identity matrix of order n
$\otimes$		Kronecker product
$\vec{W}$	$=$	$[w_{11},\ldots,w_{n1},w_{12},\ldots,w_{n2},\ldots,w_{nm}]^T \in \mathbb{R}^{n\cdot m}$, $W \in \mathbb{R}^{n\times m}$
$v = (v_i)_{i=1}^n$	$=$	$[v_1,\ldots,v_n]^T$
M'_{ij}		the ith row and the jth column of $M \in \mathbb{R}^{n\times m}$ are omitted
$M'_{i\cdot}$		the ith row of $M \in \mathbb{R}^{n\times m}$ is omitted
$M(k,:)$		the kth row of $M \in \mathbb{R}^{n\times m}$ with $k \leq n$
$M(1:k,:)$		the upper $k \times m$ part of $M \in \mathbb{R}^{n\times m}$ with $k \leq n$
$M(:,1:k)$		the left $n \times k$ part of $M \in \mathbb{R}^{n\times m}$ with $k \leq m$

Norms and spaces

$\Vert x\Vert_2$	$=$	$\sqrt{x^H x}$, the Euclidean norm for $x \in \mathbb{C}^m$
$\Vert x\Vert_\infty$	$=$	$\max_{i\in\{1,\ldots,m\}} \vert x_i\vert$, the maximum norm for $x \in \mathbb{C}^m$
$\Vert A\Vert_2$	$=$	$\max\left\{\Vert A\,x\Vert_2 \,\middle\vert\, x \in \mathbb{C}^m,\, \Vert x\Vert_2 = 1\right\}$, the 2-induced matrix norm for $A \in \mathbb{C}^{n\times m}$
$\Vert A\Vert_F$	$=$	$(\sum_{i=1}^n \sum_{j=1}^m \vert a_{ij}\vert^2)^{1/2}$, the Frobenius norm for $A \in \mathbb{C}^{n\times m}$

$\|M\|_{op}$	=	$\sup_{\substack{v\in V\\ v\neq 0}} \frac{\|Mv\|_W}{\|v\|_V}$, where $M : V \to W$ and V, W are normed vector spaces with norms $\|\cdot\|_V$, $\|\cdot\|_W$, resp.
$\mathcal{L}_p(\mathbb{R} \to \mathbb{R}^m)$		the space of p-integrable functions $f : \mathbb{R} \to \mathbb{R}^m$, $1 \leq p < \infty$, with bounded norm:
$\|f\|_{\mathcal{L}_p(\mathbb{R}\to\mathbb{R}^m)}$	=	$\left(\int_{-\infty}^{\infty} \|f(t)\|_2^p dt\right)^{1/p}$
$\mathcal{L}_{p,\text{loc}}([0,\infty) \to \mathbb{C}^m)$		the space of locally p-integrable functions $f : [0,\infty) \to \mathbb{C}^m$ with $(\int_K \|f(t)\|_2^p)^{1/p} < \infty$ for all compact $K \subset [0,\infty)$; $1 \leq p < \infty$
$\mathcal{L}_2(\jmath\mathbb{R} \to \mathbb{C}^m)$		the space of complex-valued functions $F : \jmath\mathbb{R} \to \mathbb{C}^m$ with bounded norm:
$\|F\|_{\mathcal{L}_2(\jmath\mathbb{R}\to\mathbb{C}^m)}$	=	$\left(\int_{-\infty}^{\infty} \|F(\jmath\omega)\|_2^2 d\omega\right)^{1/2}$
$\mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^m)$		the Hardy space of all analytic $F : \mathbb{C}^+ \to \mathbb{C}^m$ with bounded norm:
$\|F\|_{\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^m)}$	=	$\sup_{\alpha>0} \left(\int_{-\infty}^{\infty} \|F(\alpha+\jmath\omega)\|_2^2 d\omega\right)^{1/2}$
$\mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^{n\times m})$		the Hardy space of all analytic $F : \mathbb{C}^+ \to \mathbb{C}^{n\times m}$ with bounded norm:
$\|F\|_{\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^{n\times m})}$	=	$\sup_{\alpha>0} \left(\int_{-\infty}^{\infty} \|F(\alpha+\jmath\omega)\|_F^2 d\omega\right)^{1/2}$
$\mathcal{H}_\infty(\mathbb{C}^+ \to \mathbb{C}^m)$		the Hardy space of all analytic $F : \mathbb{C}^+ \to \mathbb{C}^m$ with bounded norm:
$\|F\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^m)}$	=	$\sup_{s\in\mathbb{C}^+} \|F(s)\|_\infty$
$\mathcal{H}_\infty(\mathbb{C}^+ \to \mathbb{C}^{n\times m})$		the Hardy space of all analytic $F : \mathbb{C}^+ \to \mathbb{C}^{n\times m}$ with bounded norm:
$\|F\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^{n\times m})}$	=	$\sup_{s\in\mathbb{C}^+} \|F(s)\|_2$
$\ell_2(\mathbb{N} \to \mathbb{R}^m)$		the space of sequences $f : \mathbb{N} \to \mathbb{R}^m$ with bounded norm:
$\|f\|_{\ell_2(\mathbb{N}\to\mathbb{R}^m)}$	=	$\left(\sum_{k\in\mathbb{N}} \|f(k)\|_2^2\right)^{1/2}$
$h_2(\mathcal{D}^+ \to \mathbb{C}^m)$		the Hardy space of all analytic $F : \mathcal{D}^+ \to \mathbb{C}^m$ with bounded norm:
$\|F\|_{h_2(\mathcal{D}^+\to\mathbb{C}^m)}$	=	$\sup_{\|r\|>1} \left(\int_{-\pi}^{\pi} \|F(re^{\jmath\theta})\|_F^2 d\theta\right)^{1/2}$
$h_\infty(\mathcal{D}^+ \to \mathbb{C}^m)$		the Hardy space of all analytic $F : \mathcal{D}^+ \to \mathbb{C}^m$ with bounded norm:
$\|F\|_{h_\infty(\mathcal{D}^+\to\mathbb{C}^m)}$	=	$\sup_{z\in\mathcal{D}^+} \|F(z)\|_\infty$
$h_\infty(\mathcal{D}^+ \to \mathbb{C}^{n\times m})$		the Hardy space of all analytic $F : \mathcal{D}^+ \to \mathbb{C}^{n\times m}$ with bounded norm:
$\|F\|_{h_\infty(\mathcal{D}^+\to\mathbb{C}^{n\times m})}$	=	$\sup_{z\in\mathcal{D}^+} \|F(z)\|_2$
$\mathcal{E}_\alpha([0,\infty) \to \mathbb{R}^m)$	=	$\left\{ f \in \mathcal{L}_{1,\text{loc}}([0,\infty) \to \mathbb{R}^m) \,\middle|\, \int_0^\infty \|f(t)\|_2 e^{-\alpha t} dt < \infty \right\}, \alpha \in \mathbb{R}$
$\mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m)$	=	$\left\{ f : \mathbb{N} \to \mathbb{R}^m \,\middle|\, \sum_{k=0}^\infty \|f(k)\| \gamma^{-k} < \infty \right\}, \gamma \in \mathbb{R}, \gamma > 0$

LTI systems	**continuous-time**
$\Sigma(A,B,C,D)$	continuous-time LTI system (2.1)
$\mathcal{P}(\cdot)$	finite controllability Gramian, see (2.10)
$\mathcal{Q}(\cdot)$	finite observability Gramian, see (2.13)
$\mathcal{P}$, $\mathcal{Q}$	infinite Gramians, see (2.15)
$\mathcal{C}_n(A,B)$	controllability or Kalman matrix of order n, see (2.11)
$\mathcal{O}_n(A,C)$	observability matrix of order n, see (2.14)
$\hat{n}$	Mc Millan degree of an LTI system, see Def. 2.1.3
$\phi(t;x_0,u(\cdot))$	solution of state equation (2.1a) at time t with initial state x_0 and input u
$U(s)=(\mathcal{L}u)(s)$	Laplace transform of a function $u(\cdot):[0,\infty)\to\mathbb{R}^m$, see Def. 2.1.18
$\mathcal{G}$	: $\mathcal{L}_{2,\mathrm{loc}}([0,\infty)\to\mathbb{R}^m)\to\mathcal{L}_{2,\mathrm{loc}}([0,\infty)\to\mathbb{R}^p)$, see (2.3)
$\mathbb{G}$	: $\mathcal{L}_2([0,\infty)\to\mathbb{R}^m)\to\mathcal{L}_2([0,\infty)\to\mathbb{R}^p)$, input-output operator, see (2.18)
G_M	$\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^m)\to\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^p)$, multiplication operator
$G(s)$	transfer function, see (2.20)
$\lvert G(\jmath\omega)\rvert$	gain of an LTI system
$\arg G(\jmath\omega)$	phase of an LTI system

LTI systems	**discrete-time**, see Section 2.2
$\Sigma(A,B,C,D)$	discrete-time LTI system (2.23)
$(\mathbb{R}^m)^{\mathbb{N}}$	space of sequences, $u(\cdot)\in(\mathbb{R}^m)^{\mathbb{N}}\Leftrightarrow u(\cdot):\mathbb{N}\to\mathbb{R}^m$
$\mathcal{P}(\cdot)$	finite controllability Gramian
$\mathcal{Q}(\cdot)$	finite observability Gramian
$\mathcal{P}$, $\mathcal{Q}$	infinite Gramians, see (2.24)
$\phi(k;x^0,u(\cdot))$	solution of state equation (2.23a) at time k with initial state x^0, input u
$\hat{u}(z)=(\mathcal{Z}u)(z)$	z-transform of a sequence $u(\cdot)\in(\mathbb{R}^m)^{\mathbb{N}}$, see Def. 2.2.6
$\mathcal{G}$	: $(\mathbb{R}^m)^{\mathbb{N}}\to(\mathbb{R}^p)^{\mathbb{N}}$, input-output operator
$\mathbb{G}$	: $\ell_2(\mathbb{N}\to\mathbb{R}^m)\to\ell_2(\mathbb{N}\to\mathbb{R}^p)$, input-output operator
G_M	$h_2(\mathcal{D}^+\to\mathbb{C}^m)\to h_2(\mathcal{D}^+\to\mathbb{C}^p)$, multiplication operator
$G(z)$	transfer function

$\mathcal{H}$-matrix

$\mathcal{I}$		finite index set
$\mathcal{I} \times \mathcal{I}$		product index set
$M_{\|r\times s}$		submatrix of M, see (2.25)
$T_{\mathcal{I}}$	=	(V, E), $\mathcal{H}$-tree with vertices V and edges E, see Def. 2.3.1
$T_{\mathcal{I}\times\mathcal{I}}$	=	block $\mathcal{H}$-tree, see Def. 2.3.4
$S(v)$	=	$\{w \in V \,\|\, (v, w) \in E\}$, set of sons for a vertex $v \in V$
$S^*(v)$	=	$\{w \in V \,\|\, w \supset v\}$, descendants of $v \in V$
$\mathcal{L}(T_{\mathcal{I}})$	=	$\{v \in V \,\|\, S(v) = \emptyset\}$, set of leaves of $T_{\mathcal{I}}$
$\mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})$		set of admissible leaves of $T_{\mathcal{I}\times\mathcal{I}}$
$\mathcal{L}^-(T_{\mathcal{I}\times\mathcal{I}})$		set of inadmissible leaves of $T_{\mathcal{I}\times\mathcal{I}}$
$\operatorname{supp}(\varphi)$	=	$\overline{\{x : \varphi(x) \neq 0\}}$, support of function $\varphi(\cdot)$
$\Omega_{\mathcal{I}}$	=	$\bigcup_{i\in\mathcal{I}} \operatorname{supp}(\varphi_i)$, domain associated to $\mathcal{I}$
$\operatorname{diam}(v)$		diameter of cluster v
$\operatorname{dist}(v, w)$		distance between clusters v and w
p		depth of $T_{\mathcal{I}\times\mathcal{I}}$, see Remark 2.3.8
$n_{\min}$		number of close supports to bound (2.26)
$\mathrm{C}_{\mathrm{sep}}$		separation constant in (2.26)
C_{sp}		sparsity constant, see Def. 2.3.11
C_{id}		idempotency constant
η		parameter in admissibility condition, see (2.27)
$\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$		$\mathcal{H}$-matrix for $T_{\mathcal{I}\times\mathcal{I}}$ with maximum blockwise rank k, Def. 2.3.9
$M_{\mathcal{H}}$		hierarchical approximation of a matrix M
$\mathcal{S}_{\mathrm{Rk}}$		storage requirements for an Rk-matrix
$\mathcal{S}_{\mathrm{full}}$		storage requirements for a full matrix
$\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k)$		storage requirements for an $\mathcal{H}$-matrix based on $T_{\mathcal{I}\times\mathcal{I}}$ with blockwise maximum rank k
$\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, \epsilon)$		storage requirements for an $\mathcal{H}$-matrix based on $T_{\mathcal{I}\times\mathcal{I}}$ with blockwise accuracy ϵ
$\mathcal{T}_{\tilde{k}\leftarrow k}$	:	$Rk \to R\tilde{k}$, truncation operator for Rk-matrices
$\mathcal{T}_{\mathcal{H},\tilde{k}\leftarrow k}$	:	$\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) \to \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, \tilde{k})$, truncation operator, Def. 2.3.13
$\oplus$ ($\ominus$)		formatted sum (substraction), see Def. 2.3.15
$T \cdot T$		product tree
$\odot_{\mathrm{best}}$, $\odot$		(best and fast) formatted multiplication, see Def. 2.3.16
$M_{\mathcal{H}}^{-1}$		approximate inverse of a matrix M, see Algorithm 3
ϵ		blockwise accuracy using adaptive arithmetic, $\epsilon \in (0, 1)$
$k(\epsilon, M_{\|r\times s})$		blockwise rank determined by ϵ, see (2.29)

Chapter 1

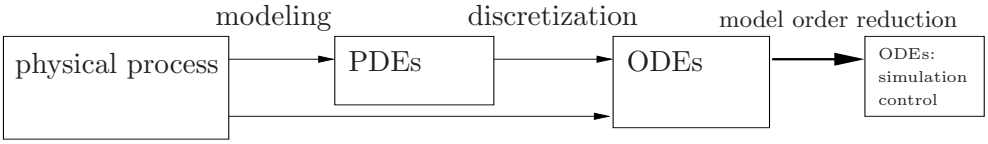
Introduction

In this thesis, efficient implementations for model order reduction of certain large-scale, linear, time-invariant (LTI) systems of the form

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t) \end{aligned}$$

are investigated. The vector $x(t) \in \mathbb{R}^n$ is referred to as the state of the system, $u(t)$ is the vector-valued input, $y(t)$ the vector-valued output, at any given time t. Alternatively, LTI systems discrete in time are considered. The dimension n of the state vector denotes the complexity or order of the system. An LTI system is called a *large-scale* system if it consists of at least hundreds of thousands of equations, i.e. $n = \mathcal{O}(10^5)$.

Large and data-sparse systems
Linear, large-scale systems arise in modeling of (complex) physical processes, such as the heat equation and linear convection-diffusion equations, with high accuracy. An additional source for high complexity is the spatial discretization of parabolic partial differential equations (PDEs) in three or more dimensions. This leads to a system of ordinary differential equations (ODEs) where the number of equations depends on the quality of the discretization. If the finite element method (FEM) is used for the spatial discretization, then n is typically large and the system matrix $A \in \mathbb{R}^{n \times n}$ is sparse; if the boundary element method (BEM) is applied, then A is fully populated but often allows for a *data-sparse* representation [89, 147, 148]. In 1998, Hackbusch [89] has introduced the hierarchical ($\mathcal{H}$) matrix format taylored for a data-sparse matrix representation. This format is shown to be sufficiently accurate for the class of matrices mentioned above and is thus suitable for the representation of many practically relevant systems. Exploiting the special structure of the hierarchical matrices in computational methods yields decreased time and memory requirements. Furthermore, Grasedyck et al. [79] proved that solutions of algebraic Riccati equations can be approximated by $\mathcal{H}$-matrices. This motivates the use of the hierarchical matrix format and the corresponding approximate arithmetic in the forthcoming control objectives.



Model order reduction

Model order reduction becomes of more fundamental importance for large-scale systems, as the computational complexity of most tasks in control including linear-quadratic optimization grows at least cubically with the order of the system. Additionally to excessive execution times, memory requirements grow quadratically and put a hard constraint on the order of the system. To make control tasks feasible for large-scale systems it is therefore necessary to reduce the dimension of the underlying system significantly. Another motivation arises in modern (LQG-, $\mathcal{H}_2$-, $\mathcal{H}_\infty$-) feedback control where the controller has at least the dimension of the underlying discretized parabolic control system. Therefore, real-time control is only possible for systems with reduced number of states resulting in controllers of lower complexity.

Model order reduction is concerned with replacing a large-scale system of order n by a system with reduced state space dimension $r \ll n$,

$$\begin{aligned} \dot{\hat{x}}(t) &= \hat{A}\hat{x}(t) + \hat{B}u(t), \\ \hat{y}(t) &= \hat{C}\hat{x}(t) + \hat{D}u(t). \end{aligned}$$

A number of methods have been proposed for model order reduction of LTI systems and can roughly be divided into two groups. The first class of methods is based on Krylov subspaces, which can equivalently be seen as moment matching methods. These methods are computationally stable and efficient but suffer from the disadvantage that no global error bound for the difference between original and reduced-order system exists in general. Furthermore, the preservation of important system properties such as stability and passivity is not guaranteed in most implementations. The second class is based on the Hankel singular values of the LTI system. Balanced truncation (BT) [127] is the most prominent method within this class. In this work, we focus on model order reduction by BT (and related methods) since the corresponding reduced-order system has several desirable properties:

1. The stability of the original system is preserved.

2. The approximation error can be made arbitrarily small and a global, computable error bound exists.

Since model order reduction has to be applied to the original large-scale system, an efficient implementation of BT is required. The lack of computational efficiency has been regarded as the major disadvantage of BT because it requires the solution of two large-scale Lyapunov equations in the continuous-time case and of two Stein equations for discrete-time systems. (A variant of the classical BT method, the cross-Gramian approach, is based on the solution of one

Sylvester equation.) During the last two decades several approaches to the solution of large-scale matrix equations were derived. Some of the approaches are especially adapted for the purpose of model order reduction as they exploit the typical low-rank property of the solution by computing approximate low-rank solution factors. To name but a few of the more popular approaches, there are the Cholesky Factor ADI method [117, 119, 135], the cyclic low-rank Smith method [86, 135], the sign function iteration [23, 25, 31, 38], and projection-type methods [100, 102, 104, 146, 150]. Most of the methods are shown to be applicable to medium-to-large-scale systems, $\mathcal{O}(10^3) - \mathcal{O}(10^4)$, when the sparsity of the system matrix is exploited. Benner et al. [6, 31] derived parallel distributed solvers based on the sign function method and the Smith iteration which can solve even large dense problems with up to $\mathcal{O}(10^5)$ states on computers with distributed memory architecture. However, all of these methods are of cubic complexity when applied to dense problems.

Contributions of this thesis

In the present work we overcome the mentioned growth of complexity for the practically relevant class of large-scale, data-sparse systems of order about $\mathcal{O}(10^5)$. In classical literature on model order reduction typically only systems of order at most a few hundreds are considered, often with the aim of admitting rather expensive algorithms (such as LMI-based optimization techniques) for systems of moderate size. We break the "curse of dimensionality" in the first stage of BT and related methods by taking additional information into account which is provided by the underlying system and exploited by the hierarchical matrix format. We develop data-sparse solvers for different types of large-scale matrix equations as

- Lyapunov equations,
- generalized Lyapunov equations,
- Stein equations,
- Sylvester equations,
- algebraic Bernoulli equations.

The algorithms compute approximate low-rank solution factors to high accuracy and within low execution time. Using these solvers as basic building blocks, efficient implementations of the following methods for model order reduction are derived:

- balanced truncation for continuous- and discrete-time systems (based on the data-sparse solvers for Lyapunov and Stein equations, respectively),
- singular perturbation approximation (SPA) to obtain zero steady-state error (based on the data-sparse solvers for Lyapunov and Stein equations),
- a cross-Gramian approach for single-input, single-output systems and for symmetric systems (based on the data-sparse solver for Sylvester equations),

- balancing-related model order reduction for unstable systems (based on the data-sparse solvers for algebraic Bernoulli and for Lyapunov equations).

These algorithms are applied to systems coming from FEM and BEM discretizations of two- and three-dimensional parabolic PDEs also including varying diffusion coefficients and convective terms. It is shown, both theoretically and in the numerical simulations, that the error between the original and the reduced-order system introduced by using the $\mathcal{H}$-matrix format is bounded. Finally, an approach for solving an optimal control problem with inequality constraints for the control is presented. The underlying parabolic PDE is discretized in space and in time. After reducing the large state space dimension of the discrete-time LTI system by the approximate BT method, standard optimization algorithm can be applied to solve the optimization problem.

The thesis is structured as follows.

Chapter 2 provides the system theoretical background of model order reduction by BT with some special attention given to norms. We also recall some basic information about the hierarchical matrix format and the corresponding formatted arithmetic.

In Chapter 3 we review the original matrix sign function and the squared Smith iteration for the solution of Lyapunov equations in continuous and discrete time, respectively. Some important topics are discussed in more detail such as scaling strategies, employing rank-revealing QR factorizations and the complexity of each iteration scheme.

In Chapter 4 we modify the algorithms from Chapter 3 using data-sparse approximations and the formatted $\mathcal{H}$-matrix arithmetic. The original cubic complexity of the iterative schemes is reduced to linear-polylogarithmic complexity. Error bounds for the forward errors in the perturbed iterates are derived. The efficiency and accuracy of the developed algorithms is illustrated by several examples and compared to solvers in usual dense arithmetic.

In Chapter 5 we develop efficient implementations of balancing-related methods for model order reduction. Besides an approximate BT method, an implementation of SPA is described and a method based on approximate low-rank factors of the cross-Gramian is proposed. Some accuracy results are presented and several numerical simulations demonstrate the performance of the new methods.

As further applications for the derived methods we discuss an approach for model order reduction of unstable systems and a two-dimensional optimal control problem with inequality control constraints in Chapter 6.

Chapter 7 offers some conclusions and briefly discusses possibilities for improvements and future research.

Chapter 2

Basics

In this chapter we collect some important concepts of system and control theory which will be needed later for model reduction by balanced truncation and related methods. We introduce basic facts about linear, finite-dimensional systems mainly following [2, 98] thereby concentrating on continuous-time systems. The main differences to the discrete-time case are pointed out at the end of the chapter.

2.1 Linear Continuous-Time Systems

We consider a *finite-dimensional, linear, time-invariant* system on $[0, \infty)$ in *standard state space form*. This is in the *continuous-time* case a differential equation, the *input-to-state equation* (or short state equation) (2.1a), combined with an algebraic equation, the *output equation* (2.1b),

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad x(0) = x_0, \tag{2.1a}$$
$$y(t) = Cx(t) + Du(t), \tag{2.1b}$$

with constant matrices $A \in \mathbb{R}^{n\times n}$ (the *state matrix*), $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, and $x_0 \in \mathbb{R}^n$. We denote the system (2.1) by $\Sigma(A, B, C, D)$. The vector $u(t) \in \mathbb{R}^m$ contains the input variables, $y(t) \in \mathbb{R}^p$ the system output, and $x(t) \in \mathbb{R}^n$ is the vector of state variables. If the input and output space are one-dimensional the system is called single-input, single-output (SISO) system. For larger dimensions $m > 1, p > 1$ we refer to (2.1) being a multi-input, multi-output (MIMO) system.

Definition 2.1.1 (Order of $\Sigma(A, B, C, D)$) *The dimension n of the matrix A in (2.1) is called the* order *(or complexity) of the corresponding system.*

We will further investigate linear, time-invariant systems of the following form,

$$E\dot{x}(t) = Ax(t) + Bu(t), \qquad x(0) = x_0, \tag{2.2a}$$
$$y(t) = Cx(t) + Du(t), \tag{2.2b}$$

for $t \geq 0$, where A, $E \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, and $x_0 \in \mathbb{R}^n$, see [57]. Linear dynamical systems, where the input-to-state equation is written with a leading matrix E, are called *descriptor systems* or *generalized state space systems*. For E being nonsingular (2.2a) can be transformed into (2.1a) by multiplication by E^{-1} from the left. We also derive algorithms which can be applied directly to systems of the form (2.2). In the following, we assume that A as well as E are nonsingular, the pencil $\lambda E - A$ is regular, i.e., $\det(\lambda E - A) \neq 0$ for some $\lambda \in \mathbb{C}$. Furthermore, $\lambda E - A$ is called a stable matrix pencil if it is regular and all finite eigenvalues of $\lambda E - A$ are in the open left half plane, i.e. $\{\lambda \in \mathbb{C} \,|\, \det(\lambda E - A) = 0\} \subset \mathbb{C}^-$. In the following we recall system theoretical concepts for systems in standard form (2.1). For more information about descriptor systems we refer to [57, 110, 158].

It is immediate, that for a given input $u(\cdot) \in \mathcal{L}_{2,\mathrm{loc}}([0,\infty) \to \mathbb{R}^m)$ and an initial state $x_0 \in \mathbb{R}^n$ at time 0, the unique solution $x(\cdot)$ of (2.1a) is given by

$$\phi(t; x_0, u(\cdot)) := x(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \quad \text{for all } t \geq 0.$$

Definition 2.1.2 (Realization) *$\Sigma(A, B, C, D)$ is called a* realization *of*

$$\begin{array}{rccl} \mathcal{G}: & \mathcal{L}_{2,loc}([0,\infty) \to \mathbb{R}^m) & \to & \mathcal{L}_{2,loc}([0,\infty) \to \mathbb{R}^p) \\ & u(\cdot) & \mapsto & y(\cdot), \end{array}$$

if there exists an initial value $x_0 \in \mathbb{R}^n$ such that for all $u(\cdot) \in \mathcal{L}_{2,loc}([0,\infty) \to \mathbb{R}^m)$:

$$\mathcal{G}u(\cdot) = Ce^{A\cdot}x_0 + \int_0^{\cdot} Ce^{A(\cdot-\tau)}Bu(\tau)d\tau + Du(\cdot). \tag{2.3}$$

Infinitely many realizations of $\Sigma(A, B, C, D)$ are obtained by a change of coordinates $\tilde{x} = Tx$, with $\det(T) \neq 0$. The system (2.1) is transformed to

$$\begin{aligned} \dot{\tilde{x}}(t) &= TAT^{-1}\tilde{x}(t) + TBu(t), & Tx(0) &= \tilde{x}_0, \\ y(t) &= CT^{-1}\tilde{x}(t) + Du(t). \end{aligned}$$

Using the notations

$$\tilde{A} = TAT^{-1}, \quad \tilde{B} = TB, \quad \tilde{C} = CT^{-1}, \quad \tilde{D} = D, \tag{2.5}$$

it follows that

$$\mathcal{G}u(\cdot) = \tilde{C}e^{\tilde{A}\cdot}\tilde{x}_0 + \int_0^{\cdot} \tilde{C}e^{\tilde{A}(\cdot-\tau)}\tilde{B}u(\tau)d\tau + \tilde{D}u(\cdot).$$

Definition 2.1.3 (Minimal realization) *A realization $\Sigma(A, B, C, D)$ of order $\hat{n}$ of $\mathcal{G}$ is called* minimal *if $\hat{n}$ is the smallest possible dimension under all possible realizations. This minimal number of states is called the* McMillan degree *of the realization.*

For the notions of stability and controllability, we can restrict ourselves to the state equation of the system. This will be expressed using the short notation $\Sigma(A, B)$.

2.1.1 Stability

Definition 2.1.4 (Hurwitz stability) *A matrix $A \in \mathbb{R}^{n\times n}$ is called* Hurwitz *(stable) if all eigenvalues of A are in the open left half of the complex plane, i.e., $\operatorname{Re}(\lambda) < 0$ for all $\lambda \in \Lambda(A)$.*

In the following we call a system $\Sigma(A, B)$ stable if A is Hurwitz.

Example 2.1.5 As an introductory example we consider the heat flow in a thin rod of length 1. The temperature $\mathbf{x}(t, \xi)$ at time $t \in (0, \infty)$ and place $\xi \in (0, 1)$ is given by the instationary heat equation

$$\frac{\partial \mathbf{x}}{\partial t}(t, \xi) = \frac{\partial^2 \mathbf{x}(t, \xi)}{\partial \xi^2}, \qquad \xi \in \Omega := (0, 1). \tag{2.6}$$

We take influence on the temperature distribution by heating or cooling at one endpoint with a heat source $u(\cdot)$:

$$\mathbf{x}(t, 1) = u(t), \qquad \text{for } t > 0.$$

At the endpoint 0 we fix the temperature by a homogeneous Dirichlet boundary condition

$$\mathbf{x}(t, 0) = 0, \qquad t > 0,$$

and at time $t = 0$ we specify an initial temperature distribution

$$\mathbf{x}(0, \xi) = \mathbf{x}_0(\xi), \qquad \xi \in \Omega.$$

For the space discretization the interval Ω is partitioned into $n + 1$ pieces of length $h = 1/(n+1)$. We want to compute the temperature at the n inner grid points

$$\xi_j = jh, \quad j = 1, \ldots, n.$$

By use of piecewise linear FEM ansatz functions with $h > 0$

$$\varphi_j(\xi) = \begin{cases} \frac{\xi - \xi_{j-1}}{h}, & \xi_{j-1} < \xi \le \xi_j, \\ \frac{\xi_{j+1} - \xi}{h}, & \xi_j < \xi < \xi_{j+1}, \\ 0, & \text{otherwise}, \end{cases}$$

for $j = 1, \ldots, n$, and a Galerkin approach with ansatz

$$\mathbf{x}(t, \xi) \approx \sum_{j=1}^{n} x_j(t)\varphi_j(\xi),$$

we obtain, by substituting into (2.6), a system of n linear first-order ordinary differential equations

$$\sum_{j=1}^{n} \underbrace{\int_0^1 \varphi_i(\xi)\varphi_j(\xi)\, d\xi}_{e_{ij}} \frac{\partial x_j}{\partial t}(t) = \sum_{j=1}^{n} \underbrace{\int_0^1 \varphi_i'(\xi)\varphi_j'(\xi)\, d\xi}_{-\hat{a}_{ij}} x_j(t), \quad \text{for } i = 1, \ldots, n.$$

The initial value is given by

$$x_j(0) = \int_0^1 \mathbf{x}_0(\xi)\varphi_j(\xi)\,d\xi, \quad \text{for } j = 1, \dots, n. \tag{2.7}$$

Rewriting the system in matrix form yields the approximate state equation in generalized state space form

$$E\dot{x}(t) = -\hat{A}x(t) + \hat{B}u(t) \tag{2.8}$$

with mass matrix $E \in \mathbb{R}^{n\times n}$, stiffness matrix $\hat{A} \in \mathbb{R}^{n\times n}$ and vector $\hat{B} \in \mathbb{R}^n$ given for exact integration of the integrals above by

$$E = \frac{h}{6}\begin{bmatrix} 4 & 1 & & & \mathbf{0} \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ \mathbf{0} & & & 1 & 4 \end{bmatrix}, \hat{A} = \frac{1}{h}\begin{bmatrix} 2 & -1 & & & \mathbf{0} \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ \mathbf{0} & & & -1 & 2 \end{bmatrix}, \hat{B} = \frac{1}{h}\begin{bmatrix} 0 \\ \\ \vdots \\ 0 \\ 1 \end{bmatrix}.$$

The eigenvalues λ_j of the symmetric, positive definite matrices $\hat{A}$ and E are given by

$$\lambda_j^E = \frac{h}{3}(2 + \cos(j\pi h)), \quad \lambda_j^{\hat{A}} = \frac{2}{h}(1 - \cos(j\pi h)), \quad j = 1, \dots, n.$$

Both matrices have the same set of eigenvectors

$$v_j = \left(\sqrt{2h}\sin(ij\pi h)\right)_{i=1}^n, \quad j = 1, \dots, n,$$

see, e.g., [141, Lemma 9.11]. For the transformed system obtained by left multiplication with the inverse of E

$$\dot{x}(t) = \underbrace{-E^{-1}\hat{A}}_{A}x(t) + \underbrace{E^{-1}\hat{B}}_{B}u(t) \tag{2.9}$$

we conclude that the eigenvalues are given by

$$\lambda_j^A = -\frac{\lambda_j^{\hat{A}}}{\lambda_j^E}, \quad \text{for } j = 1, \dots, n,$$

with all eigenvalues having negative real part:

$$-\frac{12}{h^2} < \lambda_j^A < 0.$$

Thus, the matrix A is Hurwitz. □

A weaker concept than stability is the following.

Definition 2.1.6 (Stabilizability of $\Sigma(A, B)$) *The system $\Sigma(A, B)$ is* stabilizable *if there exists an input function $u(\cdot) \in \mathcal{L}_{2,loc}([0, \infty) \to \mathbb{R}^m)$ such that the corresponding solution of (2.1a) satisfies $\lim_{t\to\infty} \phi(t; 0, u(\cdot)) = 0$.*

In the following, we restrict our attention to stable systems. This is justified since a large class of practically relevant systems, e.g. arising from the discretization of parabolic partial differential equations like the heat equation or linear convection-diffusion equations, typically possesses this property.

2.1.2 Controllability and Observability

The controllability property for a dynamical system means that for a given state x_1 there exists a control $u(\cdot)$ which steers the system from any initial state x_0 to the desired state x_1 in time t_1.

Definition 2.1.7 (Controllability) *A system $\Sigma(A, B)$ is* controllable *if for any initial and any final state, $x(0) = x_0 \in \mathbb{R}^n$ and $x_1 \in \mathbb{R}^n$, respectively, there exist a time $t_1 > 0$ and an input $u(\cdot) \in \mathcal{L}_{2,loc}([0, t_1) \to \mathbb{R}^m)$ such that*

$$\phi(t_1; x_0, u(\cdot)) = x(t_1) = x_1.$$

We consider some characterizations of controllability.

Theorem 2.1.8 [2, Section 4.2.1] *The following statements are equivalent for a system (2.1a):*

1. *$\Sigma(A, B)$ is controllable.*
2. *The* finite controllability Gramian *$\mathcal{P}$ at time $t > 0$, defined as*

$$\mathcal{P}(t) := \int_0^t e^{A\tau} B B^T e^{A^T \tau} d\tau, \tag{2.10}$$

   is positive definite for some $t > 0$.
3. *The controllability matrix (Kalman matrix)*

$$\mathcal{C}_n(A, B) = [B \quad AB \quad A^2B \quad \dots \quad A^{n-1}B] \tag{2.11}$$

   has full row rank:

$$\operatorname{rank}(\mathcal{C}_n(A, B)) = n.$$

4. *The matrix $[\lambda I - A, B]$ has full row rank for all λ in $\mathbb{C}$.*

Example 2.1.9 [Example 2.1.5 ctd.] The controllability property of our example $\Sigma(A, B)$ in (2.9) will be checked by use of Condition 4 in Theorem 2.1.8. Since controllability of (2.9) is equivalent to controllability of the generalized equation (2.8) we examine the rank of the matrix $[\lambda E + \hat{A}, \hat{B}]$ instead of rank $([\lambda I - A, B])$. Thus, if we can show that

$$\operatorname{rank}\left([\lambda E + \hat{A}, \hat{B}]\right) = n, \qquad \text{for all } \lambda \in \mathbb{C} \tag{2.12}$$

the system $\Sigma(A, B)$ as well as the system $\Sigma(E, -\hat{A}, \hat{B})$ is controllable. From the definition of $\hat{B}$ it follows that (2.12) is equivalent to

$$\operatorname{rank}\left([\lambda E + \hat{A}]'_{n\cdot}\right) = n - 1, \quad \text{with } [\lambda E + \hat{A}]'_{n\cdot} \in \mathbb{R}^{n-1 \times n}, \quad \text{for all } \lambda \in \mathbb{C}.$$

Setting $\lambda = \frac{6}{h^2}$ we obtain

$$\operatorname{rank}\left([\lambda E + \hat{A}]'_{n\cdot}\right) = \operatorname{rank}\left(\frac{1}{h}\begin{bmatrix} 6 & 0 & & & \mathbf{0} \\ 0 & 6 & & & \\ & & \ddots & & \\ & & & 6 & 0 \\ \mathbf{0} & & & & \end{bmatrix}\right) = n-1$$

and for $\lambda = -\frac{3}{h^2}$

$$\operatorname{rank}\left([\lambda E + \hat{A}]'_{n\cdot}\right) = \operatorname{rank}\left(-\frac{h}{2}\begin{bmatrix} 0 & 3 & & & \mathbf{0} \\ 3 & 0 & 3 & & \\ & \ddots & \ddots & \ddots & \\ & & 3 & 0 & 3 \\ \mathbf{0} & & & & \end{bmatrix}\right) = n-1.$$

For $\lambda \notin \{\frac{6}{h^2}, -\frac{3}{h^2}\}$ we can easily see that for $x \in \mathbb{C}^{n-1}$ with

$$\mathbf{x}^T[\lambda E + \hat{A}]'_{n\cdot} = \mathbf{0}$$

or in more detail

$$\begin{aligned} & x_1[\lambda\frac{2h}{3} + \frac{2}{h}, \frac{h}{6} - \frac{1}{h}, 0, \cdots, 0] \\ + \quad & x_2[\frac{h}{6} - \frac{1}{h}, \lambda\frac{2h}{3} + \frac{2}{h}, \frac{h}{6} - \frac{1}{h}, 0, \cdots, 0] \\ \vdots & \\ + \quad & x_{n-1}[0, \cdots, 0, \frac{h}{6} - \frac{1}{h}, \lambda\frac{2h}{3} + \frac{2}{h}, \frac{h}{6} - \frac{1}{h}] \\ = \quad & [0, \cdots, 0] \end{aligned}$$

it follows that $\mathbf{x} = \mathbf{0}$. This completes the proof. □

We turn now to the dual concept of observability. Observability is concerned with the question how to reconstruct a state $x(t_0)$ from observations $y(\cdot)$ over $[t_0, t_1]$. Without loss of generality assume that $t_0 = 0$.

Definition 2.1.10 (Observability) *A system $\Sigma(A, B, C, D)$ is said to be* observable *if for any $t_1 > 0$ the initial state $x(0)$ can be determined from the input $u(\cdot)$ and from $y(\cdot)$ in $[0, t_1]$.*

Remark 2.1.11 The output equation (2.1b) for an initial state $x(0)$ and an input signal $u(\cdot)$ is given by

$$y(t) = C\phi(t; x(0), 0) + C\phi(t; 0, u(\cdot)) + Du(t), \quad t \geq 0.$$

Since in this equation the latter two terms are known for a given input $u(\cdot)$ and $t \geq 0$, the observability problem reduces to determining $x(0)$ from $y(\cdot) = C\phi(\cdot; x(0), 0)$ (assuming $u(\cdot) \equiv 0$) over $[0, t_1]$. □

Since controllability and observability are dual concepts, it turns out that the observability characterizations are dual to the controllability characterizations in Theorem 2.1.8.

Theorem 2.1.12 [2, Section 4.2.2] *The following statements are equivalent for a system (2.1):*

1. $\Sigma(A, B, C, D)$ *is observable.*
2. *The* finite observability Gramian $\mathcal{Q}$ *at time* $t > 0$*, defined as follows*

$$\mathcal{Q}(t) := \int_0^t e^{A^T\tau} C^T C e^{A\tau} d\tau, \tag{2.13}$$

   is positive definite for some $t > 0$*.*
3. *The observability matrix*

$$\mathcal{O}_n(A, C) = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{2.14}$$

   has full column rank:

$$\operatorname{rank}\left(\mathcal{O}_n(A, C)\right) = n.$$

4. *The matrix* $\begin{bmatrix} \lambda I - A \\ C \end{bmatrix}$ *has full column rank for all* $\lambda \in \mathbb{C}$*.*

Remark 2.1.13 A weaker concept than observability is *detectability*. The concept of detectability is dual to stabilizability. Thus, $\Sigma(A, C)$ is detectable if $\Sigma(A^T, C^T)$ is stabilizable. □

For stable systems, the Gramians (2.10) and (2.13) are also defined for $t = \infty$.

Definition 2.1.14 (Infinite Gramians) *For* $\Sigma(A, B, C, D)$ *being stable, the* infinite Gramians $\mathcal{P}$ *and* $\mathcal{Q}$ *are defined as*

$$\mathcal{P} := \int_0^\infty e^{A\tau} B B^T e^{A^T\tau} d\tau, \qquad \mathcal{Q} := \int_0^\infty e^{A^T\tau} C^T C e^{A\tau} d\tau. \tag{2.15}$$

Proposition 2.1.15 [2, Proposition 4.27] *The Gramians* $\mathcal{P}$ *and* $\mathcal{Q}$ *are given for stable systems (2.1) by the unique, positive semi-definite solutions of the following (continuous-time) Lyapunov equations*

$$A\mathcal{P} + \mathcal{P}A^T + BB^T = 0, \qquad A^T\mathcal{Q} + \mathcal{Q}A + C^T C = 0,$$

respectively.

For systems which are controllable and observable, the Gramians $\mathcal{P}$ and $\mathcal{Q}$ are known to be positive definite. Systems with this property are shown to be minimal.

Proposition 2.1.16 [154, Theorem 27] *A realization $\Sigma(A, B, C, D)$ is minimal, if and only if, it is controllable and observable.*

The Gramians $\mathcal{P}$ and $\mathcal{Q}$ are not input-output invariants as they transform contragrediently under a change of state space coordinates. This can be shown by the following computations. We apply a nonsingular state space transformation T to (2.1). The Gramians $\tilde{\mathcal{P}}$ and $\tilde{\mathcal{Q}}$ of the transformed systems are given by the solution of the following transformed Lyapunov equations:

$$0 = TAT^{-1}\tilde{\mathcal{P}} + \tilde{\mathcal{P}}T^{-T}A^TT^T + TBB^TT^T = AT^{-1}\tilde{\mathcal{P}}T^{-T} + T^{-1}\tilde{\mathcal{P}}T^{-T}A^T + BB^T,$$

$$0 = T^{-T}A^TT^T\tilde{\mathcal{Q}} + \tilde{\mathcal{Q}}TAT^{-1} + T^{-T}C^TCT^{-1} = A^TT^T\tilde{\mathcal{Q}}T + T^T\tilde{\mathcal{Q}}TA + C^TC.$$

By the uniqueness of the Gramians we conclude

$$\tilde{\mathcal{P}} = T\mathcal{P}T^T, \quad \tilde{\mathcal{Q}} = T^{-T}\mathcal{Q}T^{-1}.$$

The product of the two Gramians undergoes a similarity transformation:

$$\tilde{\mathcal{P}}\tilde{\mathcal{Q}} = T\mathcal{P}\mathcal{Q}T^{-1}.$$

Thus, the eigenvalues of $\mathcal{P}\mathcal{Q}$ are input-output invariants. The positive square roots of the eigenvalues of $\mathcal{P}\mathcal{Q}$,

$$\sigma_i = \sqrt{\lambda_i(\mathcal{P}\mathcal{Q})}, \qquad i = 1, \ldots, n,$$

are called the *Hankel singular values* (HSVs) of the system (2.1).

Questions concerning the amount of energy which is needed for the control and the observation of an LTI system can be answered by help of the Gramians. In the following, the energy of a signal is interpreted as its $\mathcal{L}_2$-norm.

Proposition 2.1.17 [2, Proposition 4.12] *The minimum $\mathcal{L}_2$-energy required to steer a controllable system $\Sigma(A, B)$ from 0 to x_1 in $[0, t_1]$ is given by the $\mathcal{L}_2$-energy of the input function*

$$u(t) = B^Te^{A^T(t_1-t)}\xi, \qquad \xi \in \mathbb{R}^n,\ t \in [0, t_1], \tag{2.16}$$

i.e. by the square root of

$$\|u\|_{\mathcal{L}_2}^2 = \int_0^{t_1} \xi^Te^{A(t_1-t)}BB^Te^{A^T(t_1-t)}\xi\, dt = \xi^T\mathcal{P}(t_1)\xi = x_1^T\mathcal{P}(t_1)^{-1}x_1,$$

where $x_1 = \phi(t_1; 0, u(\cdot)) = \mathcal{P}(t_1)\xi$. Furthermore, since from (2.10) it follows that $\mathcal{P}(t_1) \geq \mathcal{P}(t_2)$ for $t_1 \geq t_2$, the square of the minimal energy for the transfer from 0 to x_1 is given for $t_1 \to \infty$ by

$$\|u\|_{\mathcal{L}_2}^2 = \int_0^{\infty} u(t)^Tu(t)\, dt = x_1^T\mathcal{P}^{-1}x_1$$

or by reversing the time interval to $(-\infty, 0]$*, setting* $x(0) = x_0$*, by*

$$\|u\|_{\mathcal{L}_2}^2 = \int_{-\infty}^{0} u(t)^T u(t)\, dt = x_0^T \mathcal{P}^{-1} x_0.$$

For an observable system, the energy of the output $y(\cdot)$ at time t_1 generated from the initial input $x(0) = x_0$ assuming $u(\cdot) \equiv 0$ can be expressed by

$$\|y\|_{\mathcal{L}_2}^2 = \int_{0}^{t_1} x_0^T e^{A^T t} C^T C e^{At} x_0\, dt = x_0^T \mathcal{Q}(t_1) x_0.$$

Then, the square of the (largest) output energy resulting from initial state x_0 and $u(\cdot) \equiv 0$ is given for an infinite observation interval by

$$\|y\|_{\mathcal{L}_2}^2 = \int_{0}^{\infty} y(t)^T y(t)\, dt = x_0^T \mathcal{Q}\, x_0.$$

The main energy E for the transfer from past inputs to future outputs can be derived by combining these expressions:

$$E := \sup_{\substack{u \in \mathcal{L}_2((-\infty,0] \to \mathbb{R}^m) \\ \|u\|_{\mathcal{L}_2} \neq 0}} \frac{\|y\|_{\mathcal{L}_2}^2}{\|u\|_{\mathcal{L}_2}^2} = \frac{x_0^T \mathcal{Q} x_0}{x_0^T \mathcal{P}^{-1} x_0} = \frac{\overline{x}_0^T \mathcal{P}^{1/2} \mathcal{Q} \mathcal{P}^{1/2} \overline{x}_0}{\overline{x}_0^T \overline{x}_0}, \tag{2.17}$$

with $\overline{x}_0 = \mathcal{P}^{1/2} x_0$.

2.1.3 Transfer Functions

Throughout this section we assume that the LTI system (2.1) is stable and $x(0) = x_0 = 0$. Hence $\mathcal{G}$ in (2.3) maps $\mathcal{L}_2$-functions to $\mathcal{L}_2$-functions and we introduce

$$\begin{aligned} \mathbb{G}: \ \mathcal{L}_2([0,\infty) \to \mathbb{R}^m) &\to \mathcal{L}_2([0,\infty) \to \mathbb{R}^p) \\ u(\cdot) &\mapsto \mathbb{G}u(\cdot) = y(\cdot) = \int_0^{\cdot} C e^{A(\cdot - \tau)} B u(\tau) d\tau + D u(\cdot) \end{aligned} \tag{2.18}$$

as the input-output operator of a stable LTI system with zero initial value. The Laplace transform is used for the analysis of linear time-invariant systems. It can be interpreted as a transformation from the time domain to the frequency domain.

Definition 2.1.18 (Laplace transform) *For* $f(\cdot) \in \mathcal{L}_{1,loc}([0,\infty) \to \mathbb{R}^m)$ *and* $s \in \mathbb{C}$

$$F(s) = (\mathcal{L}f)(s) := \int_0^{\infty} f(t) e^{-st} dt$$

is called the Laplace transform *of* $f(\cdot)$*, if it exists.*

Let $\alpha \in \mathbb{R}$ be fixed. Consider

$$\mathcal{E}_\alpha([0,\infty) \to \mathbb{R}^m) := \left\{ f \in \mathcal{L}_{1,\mathrm{loc}}([0,\infty) \to \mathbb{R}^m) \,\middle|\, \int_0^\infty \|f(t)\|_2 e^{-\alpha t} dt < \infty \right\}$$

and

$$\mathbb{C}_\alpha := \{s \in \mathbb{C} | \operatorname{Re}(s) > \alpha\}.$$

It is shown, e.g. in [98, page 742], that the Laplace transform $F(\cdot)$ of $f(\cdot) \in \mathcal{E}_\alpha([0,\infty) \to \mathbb{R}^m)$ is analytic and bounded on $\mathbb{C}_\alpha$, i.e.

$$\begin{array}{rcl} \mathcal{L}: \ \mathcal{E}_\alpha([0,\infty) \to \mathbb{R}^m) & \to & \mathcal{H}_\infty(\mathbb{C}_\alpha \to \mathbb{C}^m) \\ f(\cdot) & \mapsto & (\mathcal{L}f)(\cdot) \end{array}$$

is well defined. Since the Cauchy-Schwarz inequality gives

$$\mathcal{L}_2([0,\infty) \to \mathbb{R}^m) \subset \mathcal{E}_\alpha([0,\infty) \to \mathbb{R}^m) \quad \text{for all } \alpha > 0,$$

it follows that

$$\mathcal{L}: \ \mathcal{L}_2([0,\infty) \to \mathbb{R}^m) \to \mathcal{H}_\infty(\mathbb{C}^+ \to \mathbb{C}^m).$$

As in [98, Theorem A.3.47] one can show that

$$(2\pi)^{-1/2}\mathcal{L}: \ \mathcal{L}_2([0,\infty) \to \mathbb{R}^m) \to \mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^m) \tag{2.19}$$

is a bounded operator and an isometry.

Applying the Laplace transform to a system with $u(\cdot) \in \mathcal{E}_\alpha([0,\infty) \to \mathbb{R}^m)$ for some $\alpha > 0$ yields a system representation in the frequency domain

$$\begin{array}{rcl} sX(s) & = & AX(s) + BU(s), \\ Y(s) & = & CX(s) + DU(s), \quad \text{for all } s \in \mathbb{C}_\alpha. \end{array}$$

A connection between input and output variables in the frequency domain is obtained via the so called *transfer function* or *transfer function matrix* (TFM) associated with the system $\Sigma(A,B,C,D)$:

$$G(s) = C(sI - A)^{-1}B + D, \qquad s \in \overline{\mathbb{C}^+}. \tag{2.20}$$

$G(\cdot)$ is a matrix over the field of rational functions and defines the multiplication operator

$$\begin{array}{rcl} G_M: \ \mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^m) & \to & \mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^p) \\ U(\cdot) & \mapsto & Y(\cdot) = G(\cdot)\,U(\cdot), \end{array}$$

and hence

$$(\mathcal{L}y)(s) = Y(s) = \big(C(sI - A)^{-1}B + D\big)\,U(s) = G(s)(\mathcal{L}u)(s), \quad \text{for all } s \in \mathbb{C}^+ .$$

The relationship between the norm of G_M in the frequency domain and the norm of $\mathbb{G}$ in time domain is given by the isometry property of the (normalized) Laplace transform (2.19) in the following commuting diagram:

$$\begin{array}{ccc} \mathcal{L}_2([0,\infty) \to \mathbb{R}^m) & \xrightarrow{\mathbb{G}} & \mathcal{L}_2([0,\infty) \to \mathbb{R}^p) \\ (2\pi)^{-1/2}\mathcal{L} \downarrow & & \downarrow (2\pi)^{-1/2}\mathcal{L} \\ \mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^m) & \xrightarrow{G_M} & \mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^p) \end{array} .$$

It follows

$$\|\mathbb{G}\|_{\text{op}} = \sup_{\substack{u\in\mathcal{L}_2 \\ \|u\|_{\mathcal{L}_2}\neq 0}} \frac{\|\mathbb{G}u\|_{\mathcal{L}_2([0,\infty)\to\mathbb{R}^p)}}{\|u\|_{\mathcal{L}_2([0,\infty)\to\mathbb{R}^m)}} = \sup_{\substack{U\in\mathcal{H}_2 \\ \|U\|_{\mathcal{H}_2}\neq 0}} \frac{\|GU\|_{\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^p)}}{\|U\|_{\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^m)}} = \|G_M\|_{\text{op}}.$$

It is well known, as for example stated and proved in [98, Theorem 2.3.28] that the 2-induced operator norm of $\mathbb{G}$ equals the $\mathcal{H}_\infty$-norm

$$\|\mathbb{G}\|_{\text{op}} = \|G\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^{p\times m})}.$$

As a consequence we have, for all $u \in \mathcal{L}_2([0,\infty)\to\mathbb{R}^m)$, the useful estimates

$$\|y\|_{\mathcal{L}_2([0,\infty)\to\mathbb{R}^m)} \leq \|G\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^{p\times m})}\|u\|_{\mathcal{L}_2([0,\infty)\to\mathbb{R}^p)} \tag{2.21}$$

and

$$\|Y\|_{\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^p)} \leq \|G\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^{p\times m})}\|U\|_{\mathcal{H}_2(\mathbb{C}^+\to\mathbb{C}^m)}. \tag{2.22}$$

We will revisit these bounds in Chapter 5 for the important question in model order reduction of how "close" the original and the reduced-order systems are. Note that the $\mathcal{H}_\infty$-norm is endowed with the 2-induced norm:

$$\|G\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^{p\times m})} = \sup_{s\in\mathbb{C}^+} \|G(s)\|_2 = \sup_{s\in\mathbb{C}^+} \sigma_{\max}(G(s)).$$

The stability of A yields that $G(\cdot)$ is analytic on $\mathbb{C}^+$ and bounded, i.e.,

$$G(\cdot) \;\in\; \mathcal{H}_\infty(\mathbb{C}^+ \;\to\; \mathbb{C}^{p\times m}).$$

Since $G(\cdot)$ is continuous on $\overline{\mathbb{C}^+}$, the maximum modulus theorem yields that the supremum is attained on the imaginary axis, i.e.,

$$\|G\|_{\mathcal{H}_\infty(\mathbb{C}^+\to\mathbb{C}^{p\times m})} = \sup_{\omega\in\mathbb{R}} \sigma_{\max}(G(\jmath\omega)).$$

The TFM G is invariant under coordinate changes in state space. To see this we introduce new coordinates $\tilde{x} = Tx$, with $\det(T) \neq 0$, and obtain a new realization $\Sigma(\tilde{A},\tilde{B},\tilde{C},D)$ for the system (2.1) using the notations in (2.5). It is seen that the associated TFM $\tilde{G}$ equals G

$$\begin{aligned} G(s) &= C(sI-A)^{-1}B + D = CT^{-1}T(sI-A)^{-1}T^{-1}TB + D \\ &= \tilde{C}(sI-\tilde{A})^{-1}\tilde{B} + D = \tilde{G}(s). \end{aligned}$$

Let us examine the input-output behavior of an LTI system in the time domain for an exponential input and a sinusoidal and describe it via the TFM. The output of (2.1) for an exponential input $u(t) = e^{st}$, $s \in \mathbb{C}\setminus\Lambda(A)$, $t \leq 0$, can be expressed by the TFM G as follows:

$$\begin{aligned} y(t) &= Cx(t) + Du(t) = C\phi(t;x_0,e^{s\cdot}) + De^{st} \\ &= C\left(e^{At}x_0 + \int_0^t e^{A(t-\tau)}Be^{s\tau}d\tau\right) + De^{st} \\ &= C\left(e^{At}x_0 + e^{At}(sI-A)^{-1}(e^{(sI-A)t} - I)B\right) + De^{st} \\ &= Ce^{At}\left(x_0 - (sI-A)^{-1}B\right) + G(s)e^{st}. \end{aligned}$$

For A being Hurwitz, the system response to the input $u(t) = \sin(\omega t)$, $\omega \in \mathbb{R}$, is

$$y(t) = \delta(t) + |G(\jmath\omega)| \sin(\omega t + \arg G(\jmath\omega)), \quad t \geq 0,$$

where $\delta(t) \to 0$ for $t \to \infty$. Hence

$$t \mapsto |G(\jmath\omega)| \sin(\omega t + \arg G(\jmath\omega)), \quad t \geq 0,$$

is called the "steady state response" and $|G(\jmath\omega)|$ is the gain, $\arg G(\jmath\omega)$ the phase of the system [98, Proposition 2.3.22.].

For $\omega \in \mathbb{R}$ the complex-valued function $\omega \mapsto G(\jmath\omega)$ is called the *frequency response*. The frequency response can be graphically represented by two curves, the gain curve ($\omega \mapsto |G(\jmath\omega)|$) and the phase curve ($\omega \mapsto \arg G(\jmath\omega)$). An output of both curves is called a *Bode plot* if logarithmic scales are used for the frequency [130, Chapter 6]. From this plot important properties of an LTI system, as for instance gain and phase margin, DC gain, bandwidth and stability, can be analyzed over a wide range of frequencies. We discuss the results of the model reduction methods in Chapter 5 by help of the Bode magnitude plot.

Example 2.1.19 [Example 2.1.5 ctd.] As output for Example 2.1.5 we measure the temperature at the center of the rod. We obtain the output equation

$$\mathbf{x}(t, 1/2) \approx y(t) = Cx(t), \qquad t > 0,$$

with $C^T \in \mathbb{R}^n$, $C = e_{(n+1)/2}^T$ for n being odd. In Figure 2.1 the frequency response of the LTI system (2.8) with additional output as described above is plotted in a Bode diagram using the function `bode` from the MATLAB Control System Toolbox. Here, the gain $|G(\jmath\omega)|$ is plotted in decibel, $20\log_{10}(|G(\jmath\omega)|)$, and the phase in degrees. □

2.2 Linear Discrete-Time Systems

In this section we consider LTI systems which are discrete in time

$$x_{k+1} = Ax_k + Bu_k, \qquad x_0 = x^0, \tag{2.23a}$$
$$y_k = Cx_k + Du_k, \tag{2.23b}$$

with initial condition $x^0 \in \mathbb{R}^n$ and $k \in \mathbb{N}$. The dimensions of the matrices involved are equal to those in the continuous-time setting (2.1), $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, and therefore, the order of the system is n. We shortly point out the main differences to the continuous-time case. The time dependency of an element of the sequence $u(\cdot) : \mathbb{N} \to \mathbb{R}^m$ will be denoted in the following by u_k or by $u(k)$ for better readability depending on the situation. For the space of sequences $u(\cdot) : \mathbb{N} \to \mathbb{R}^m$ we also use the notation $u(\cdot) \in (\mathbb{R}^m)^{\mathbb{N}}$. The solution of the state equation (2.23a) at time $k \in \mathbb{N}$ under influence of the input $u(\cdot) \in (\mathbb{R}^m)^{\mathbb{N}}$ and with initial condition x^0 at time 0 is given by

$$\phi(k; x^0, u(\cdot)) = A^k x^0 + \sum_{j=0}^{k-1} A^{k-1-j} B u(j).$$

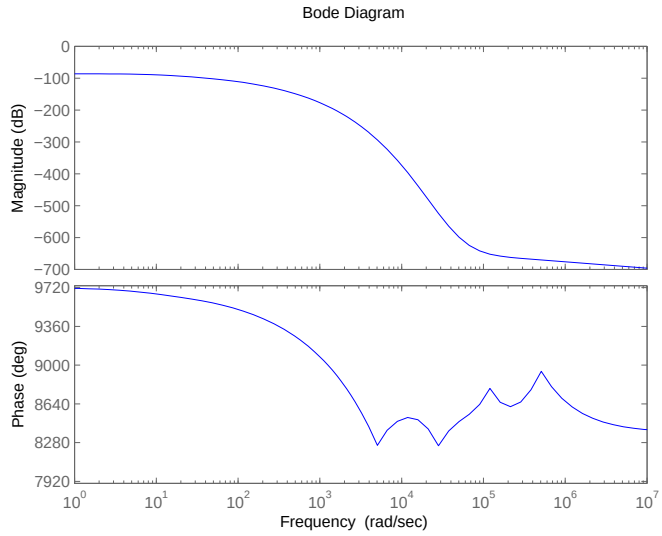


Figure 2.1: Bode diagram for Example 2.1.19.

The corresponding input-output operator of (2.23) with time domain $\mathbb{N}$ is defined by

$$\begin{aligned} \mathcal{G}: \ (\mathbb{R}^m)^{\mathbb{N}} &\to (\mathbb{R}^p)^{\mathbb{N}} \\ u(\cdot) &\mapsto y(\cdot), \end{aligned}$$

$$\mathcal{G}u(\cdot) = CA^{\cdot}x^0 + \sum_{j=0}^{\cdot-1} CA^{(\cdot-1-j)}Bu(j) + Du(\cdot).$$

Analogously to the continuous-time case we define the stability of the system.

Definition 2.2.1 (Schur Stability) *The matrix A is called* Schur stable *or* convergent *if all eigenvalues of A are in the interior of the unit disk $\mathcal{D}$, i.e., $|\lambda| < 1$ for all $\lambda \in \Lambda(A)$.*

In the following, a discrete-time system $\Sigma(A, B)$ is called stable if A is Schur stable.

Example 2.2.2 [Example 2.1.5 ctd.] We consider the state equation (2.8) of Example 2.1.5. The time interval $[0, \infty)$ is discretized with constant step size $T_s > 0$. The computed solution at the kth time-step $t_k = kT_s$ is denoted by $x_k := x(t_k)$ for $k \in \mathbb{N}$. The backward Euler method computes x_{k+1} by

$$\begin{aligned} x_{k+1} &= (E + T_s\hat{A})^{-1}E\,x_k + T_s(E + T_s\hat{A})^{-1}\hat{B}\,u_k, \\ y_k &= Cx_k, \quad \text{for } k \in \mathbb{N} \end{aligned}$$

with $x(0) = x^0$ given as in (2.7). That is, we obtain a full discretization of the control problem in Example 2.1.5 using an approximate derivative

$$\dot{x}(t_k) \approx \frac{x_{k+1} - x_k}{T_s}.$$

The eigenvalues of $(E + T_s\hat{A})^{-1}E$ are given by the eigenvalues of E and $\hat{A}$ as follows:

$$\lambda_j((E + T_s\hat{A})^{-1}E)_j = \frac{\lambda_j^E}{\lambda_j^E + T_s\lambda_j^{\hat{A}}}.$$

Thus, for arbitrary $T_s > 0$ (and $h > 0$ in the definition of E and $\hat{A}$) the matrix $(E + T_s\hat{A})^{-1}E$ is Schur stable. □

In the following we introduce the concept of reachability which is stronger than controllability. Note that for continuous-time systems these concepts coincide.

Definition 2.2.3 (Reachability) *For a system (2.23) a state $x^1 \in \mathbb{R}^n$ is called reachable from the zero state if there exist an input $u(\cdot) : \mathbb{N} \to \mathbb{R}^m$ and a time $k_1 < \infty$ such that*

$$\phi(k_1; 0, u(\cdot)) = x(k_1) = x^1.$$

The finite Gramian at time $k \in \mathbb{N} \setminus 0$ is defined by

$$\mathcal{P}(k) := \sum_{j=0}^{k-1} A^j B B^T (A^T)^j,$$

for the reachability concept and for the observability by

$$\mathcal{Q}(k) := \sum_{j=0}^{k-1} (A^T)^j C^T C A^j.$$

With these definitions the equivalences in Theorem 2.1.8 and Theorem 2.1.12 also hold for the discrete-time case.

Definition 2.2.4 (Infinite Gramians) *For stable, discrete-time systems (2.23) the infinite Gramians are defined as follows:*

$$\mathcal{P} := \sum_{j=0}^{\infty} A^j B B^T (A^T)^j, \qquad \mathcal{Q} := \sum_{j=0}^{\infty} (A^T)^j C^T C e^{A^j}. \tag{2.24}$$

Proposition 2.2.5 *The infinite Gramians $\mathcal{P}$ and $\mathcal{Q}$ of a stable system (2.23) satisfy the following discrete-time Lyapunov equations*

$$A\mathcal{P}A^T + BB^T = \mathcal{P}, \qquad A^T\mathcal{Q}A + C^TC = \mathcal{Q}.$$

These matrix equations are also called Stein equations*.*

The input of minimum ℓ_2-energy (2.16) for the controllable system (2.23a) changes in discrete time to

$$u(k) = B^T(A^T)^{(k_1-k)}\xi, \qquad k \in [0, k_1],\ k_1 \geq n,$$

for some $\xi \in \mathbb{R}^n$ such that $x_1 = \phi(k_1; 0, u(\cdot)) = \mathcal{P}(k_1)\xi$.

The discrete analogon to the Laplace transform is the z-transform. In the following we use the notation $\mathcal{D}^+ = \{z \in \mathbb{C} \,|\, |z| > 1\}$ for the complement of the closed unit disc.

Definition 2.2.6 (z-transform) *For a sequence $u : \mathbb{N} \to \mathbb{R}^m$ and $z \in \mathbb{C}$ the z-transform is defined by the formal power series in z^{-1}*

$$\hat{u}(z) = (\mathcal{Z}u)(z) := \sum_{k=0}^{\infty} u(k) z^{-k}.$$

Let $\gamma > 0$ be fixed. Consider

$$\mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m) := \left\{ u : \mathbb{N} \to \mathbb{R}^m \;\middle|\; \sum_{k=0}^{\infty} |u(k)|\, \gamma^{-k} < \infty \right\},$$

then $u(\cdot) \in \mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m)$ yields that $\hat{u} = \mathcal{Z}u$ is absolutely convergent for all $z \in \mathbb{C}$ with $|z| \geq \gamma$. For every $u(\cdot) \in \mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m)$ the z-transform $\hat{u}(\cdot)$ is continuous for all $|z| \geq \gamma$ and analytic on

$$\mathcal{D}_\gamma := \{z \in \mathbb{C} \,|\, |z| > \gamma\}$$

[98, page 737], thus

$$\begin{aligned} \mathcal{Z} :\ \mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m) &\to h_\infty(\mathcal{D}_\gamma \to \mathbb{C}^m) \\ u(\cdot) &\mapsto (\mathcal{Z}u)(\cdot) \end{aligned}$$

is well defined. If $u(\cdot) \in \ell_2(\mathbb{N} \to \mathbb{R}^m)$ it follows by the Cauchy-Schwarz inequality that $u(\cdot) \in \mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m)$ for all $\gamma > 1$, and thus

$$\mathcal{Z} :\ \ell_2(\mathbb{N} \to \mathbb{R}^m) \to h_\infty(\mathcal{D}^+ \to \mathbb{C}^m).$$

Furthermore,

$$(2\pi)^{-1/2}\mathcal{Z} :\ \ell_2(\mathbb{N} \to \mathbb{R}^m) \to h_2(\mathcal{D}^+ \to \mathbb{C}^m)$$

is a linear isometry, see [98, Theorem A.3.43].

The TFM is obtained by applying the z-transform to the stable system (2.23) with $x_0 = x^0 = 0$ and $u(\cdot) \in \mathcal{S}_\gamma(\mathbb{N} \to \mathbb{R}^m)$ for some $\gamma > 0$, yielding

$$G(z) = C(zI - A)^{-1}B + D, \qquad z \in \overline{\mathcal{D}^+}.$$

We introduce the multiplication operator

$$\begin{aligned} G_M :\ h_2(\mathcal{D}^+ \to \mathbb{C}^m) &\to h_2(\mathcal{D}^+ \to \mathbb{C}^p) \\ \hat{u}(\cdot) &\mapsto \hat{y}(\cdot) = G(\cdot)\, \hat{u}(\cdot), \end{aligned}$$

such that

$$(\mathcal{Z}y)(z) = \hat{y}(z) = \big(C(zI - A)^{-1}B + D\big)\, \hat{u}(z) = G(z)(\mathcal{Z}u)(z), \quad \text{for all } z \in \mathcal{D}^+ .$$

For stable LTI system (2.23) with $x_0 = x^0 = 0$ we further consider the input-output operator $\mathbb{G}$, defined by

$$\begin{aligned} \mathbb{G} :\ \ell_2(\mathbb{N} \to \mathbb{R}^m) &\to \ell_2(\mathbb{N} \to \mathbb{R}^p) \\ u(\cdot) &\mapsto \mathbb{G}u(\cdot) = y(\cdot) = \sum_{j=0}^{\cdot -1} CA^{\cdot -1-j}Bu(j) + Du(\cdot). \end{aligned}$$

Note that isometry of the z-transform yields that the following diagram commutes:

$$\begin{array}{ccc} \ell_2(\mathbb{N} \to \mathbb{R}^m) & \xrightarrow{\mathbb{G}} & \ell_2(\mathbb{N} \to \mathbb{R}^p) \\ (2\pi)^{-1/2}\mathcal{Z} \downarrow & & \downarrow (2\pi)^{-1/2}\mathcal{Z} \\ h_2(\mathcal{D}^+ \to \mathbb{C}^m) & \xrightarrow{G_M} & h_2(\mathcal{D}^+ \to \mathbb{C}^p) \end{array}.$$

The equivalence of the operator norms in time and in frequency domain follows

$$\|\mathbb{G}\|_{\mathrm{op}} = \sup_{\substack{u \in \ell_2 \\ u \neq 0}} \frac{\|\mathbb{G}u\|_{\ell_2(\mathbb{N}\to\mathbb{R}^p)}}{\|u\|_{\ell_2(\mathbb{N}\to\mathbb{R}^m)}} = \sup_{\substack{\hat{u} \in h_2 \\ \|\hat{u}\|_{h_2} \neq 0}} \frac{\|G\hat{u}\|_{h_2(\mathcal{D}^+\to\mathbb{C}^p)}}{\|\hat{u}\|_{h_2(\mathcal{D}^+\to\mathbb{C}^m)}} = \|G_M\|_{\mathrm{op}}.$$

By the equivalence of the 2-induced operator norm of $\mathbb{G}$ and the h_∞-norm [98, Theorem 2.3.28]

$$\|\mathbb{G}\|_{\mathrm{op}} = \|G\|_{h_\infty(\mathcal{D}^+\to\mathbb{C}^{p\times m})},$$

we have estimates in time and in frequency domain for all $u \in \ell_2(\mathbb{N} \to \mathbb{R}^m)$,

$$\|y\|_{\ell_2(\mathbb{N}\to\mathbb{R}^m)} \leq \|G\|_{h_\infty(\mathcal{D}^+\to\mathbb{C}^{p\times m})}\|u\|_{\ell_2(\mathbb{N}\to\mathbb{R}^p)}$$

and

$$\|\hat{y}\|_{h_2(\mathcal{D}^+\to\mathbb{C}^p)} \leq \|G\|_{h_\infty(\mathcal{D}^+\to\mathbb{C}^{p\times m})}\|\hat{u}\|_{h_2(\mathcal{D}^+\to\mathbb{C}^m)}.$$

The TFM is analytic on $\mathcal{D}^+$ and bounded, i.e.,

$$G(\cdot) \in h_\infty(\mathcal{D}^+ \to \mathbb{C}^{p\times m}).$$

Since $G(\cdot)$ is continuous on $\overline{\mathcal{D}^+}$ it attains its maximum at the boundary

$$\|G\|_{h_\infty(\mathcal{D}^+\to\mathbb{C}^{p\times m})} = \sup_{z\in\mathcal{D}^+} \sigma_{\max}(G(z)) = \max_{\theta\in[-\pi,\pi]} \sigma_{\max}(G(e^{\jmath\theta})).$$

2.3 $\mathcal{H}$-Matrix Arithmetic Introduction

The hierarchical ($\mathcal{H}$)-matrix format is a data-sparse representation for a special class of matrices which often arise in applications. Matrices that belong to this class result, for instance, from the discretization of partial differential or integral equations. Exploiting the special structure of these matrices in computational methods yields decreased time and memory requirements. The $\mathcal{H}$-matrix format was introduced by Hackbusch in 1998 [89]; further detailed descriptions can be found, e.g. in [46, 75, 77, 91].

We first introduce some basic definitions for the construction of such $\mathcal{H}$-matrices then we describe the corresponding approximate arithmetic, mainly following [77]. Thereby we will point out the facts needed throughout the subsequent chapters.

The basic idea of the $\mathcal{H}$-matrix format is to partition a given matrix $M \in \mathbb{R}^{n\times n}$ recursively into submatrices $M_{|_{r\times s}}$ that admit data-sparse low-rank approximations in the following form

$$M_{|_{r\times s}} \approx AB^T, \quad A \in \mathbb{R}^{r\times k},\, B \in \mathbb{R}^{s\times k}, \tag{2.25}$$

where k denotes the rank of the submatrix. The factors are assumed to be rectangular with k much smaller than r and s. To determine such a partitioning, we first consider the hierarchical splitting of a finite index set $\mathcal{I}$, where $\mathcal{I} = \{1, \ldots, n\}$ corresponds to a finite element or boundary element basis $(\varphi_i)_{i\in\mathcal{I}}$. For simplicity, we identify the vertices of the hierarchical tree with the corresponding subset of $\mathcal{I}$. This is well defined if each vertex is either a leaf or has more than one son.

Definition 2.3.1 ($\mathcal{H}$-Tree $T_\mathcal{I}$) *For a finite index set $\mathcal{I}$ we denote by $T_\mathcal{I} = (V, E)$ a tree with vertices in V and edges in E. The set of sons for a vertex $v \in V$ is defined as*

$$S(v) := \{w \in V | \, (v, w) \in E\},$$

the descendants of v by

$$S^*(v) := \{w \in V | \, w \supset v\}.$$

$T_\mathcal{I}$ is called an $\mathcal{H}$-tree *of the index set $\mathcal{I}$ if the following conditions are satisfied:*

1. *The index set $\mathcal{I}$ is the root of $T_\mathcal{I}$, and $v \subset \mathcal{I}$, $v \neq \emptyset$, for all $v \in V$.*
2. *Either a vertex is a leaf*

$$S(v) = \emptyset, \quad (\textit{set of leaves: } \mathcal{L}(T_\mathcal{I}) := \{v \in V | \, S(v) = \emptyset\})$$

or it is the disjoint union of its sons

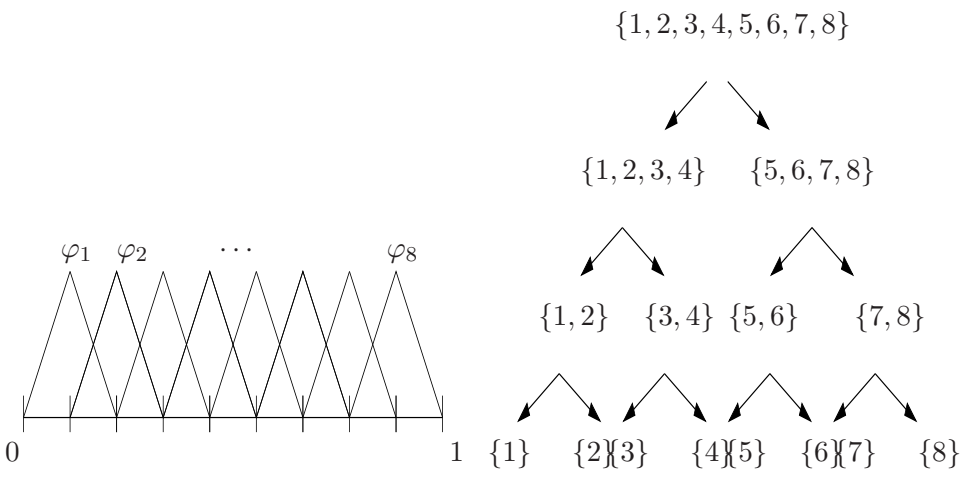
$$v = \dot{\bigcup}_{w\in S(v)} w.$$

A simple exemplary $\mathcal{H}$-tree is shown in Figure 2.2(b). For simplifying the notation we identify the set of vertices V with $T_\mathcal{I}$, omitting the set of edges. The vertices $v \in V$ are also called *clusters*.

Remark 2.3.2 (Construction of $T_\mathcal{I}$) We shortly describe the construction of a *geometrically balanced* $\mathcal{H}$-tree $T_\mathcal{I}$ by bisection. Starting with the index set $\mathcal{I}^{(0)} = \mathcal{I}$ we identify the associated domain in $\mathbb{R}^d$ by the union of the supports of the corresponding ansatz functions:

$$\Omega_{\mathcal{I}^{(0)}} := \bigcup_{i\in\mathcal{I}} \operatorname{supp}(\varphi_i), \quad \text{where} \qquad \operatorname{supp}(\varphi_i) := \overline{\{x : \, \varphi_i(x) \neq 0\}}.$$

Then, the index set $\mathcal{I}^{(0)}$ is divided into two sets $\mathcal{I}_1^{(1)}$, $\mathcal{I}_2^{(1)}$ such that the corresponding domains $\Omega_{\mathcal{I}_1^{(1)}}$ and $\Omega_{\mathcal{I}_2^{(1)}}$ have approximately equal size diameters using the Euclidean norm. The domain $\Omega_{\mathcal{I}^{(0)}}$ is thereby divided according to the space direction with maximal diameter. The splitting will be stopped as soon as the cardinality of the clusters is one. For n being a power of two, the depth of $T_\mathcal{I}$, that is the length of the longest path, is given by $p = \log_2(n)$. The computation of diameters of clusters (and of distances between them) will also be needed for the $\mathcal{H}$-matrix representation to identify matrix blocks which allow for a low-rank approximation. In practice we avoid the computation of



(a) Underlying ansatz space. (b) $\mathcal{H}$-tree $T_{\mathcal{I}}$.

Figure 2.2: Construction of a binary $\mathcal{H}$-tree $T_{\mathcal{I}}$.

these quantities for the domain Ω_v of a cluster v using instead a surrounding cuboid which is supposed to be minimal and axis parallel $Q_v \supset \Omega_v$. Using these so-called bounding boxes, the computation simplifies to

$$\operatorname{diam}(v) := \max_{x,y\in Q_v} \|x-y\|_2, \quad \operatorname{dist}(v,w) := \min_{x\in Q_v, y\in Q_w} \|x-y\|_2.$$

For index sets corresponding to regular grids it is also efficient to use a cardinality balanced splitting where the number of indices for all vertices on the same hierarchical level is approximately balanced.

□

Example 2.3.3 We consider a finite element ansatz space $(\varphi_i)_{i\in\mathcal{I}}$ as introduced in Example 2.1.5 with $n = 8$ and $\mathcal{I} = \{1,\dots,8\}$, see Figure 2.2(a). For an efficient use of the hierarchical matrix format considering both, computational complexity and storage requirements, the space of ansatz functions should be locally separated. The maximal number of functions with relatively close support is given by

$$\max_{i\in\mathcal{I}} \#\{j\in\mathcal{I} | \operatorname{dist}(\operatorname{supp}(\varphi_i),\operatorname{supp}(\varphi_j)) \le \mathrm{C_{sep}}^{-1} \operatorname{diam}(\operatorname{supp}(\varphi_i))\}. \tag{2.26}$$

For this example, setting $\mathrm{C_{sep}} = 3$, a small constant $n_{\min} = 3$ bounds the number of close supports independently of n. The $\mathcal{H}$-tree $T_{\mathcal{I}}$ in Figure 2.2(b) is constructed by bisection. In this example both splitting strategies (geometrically or cardinality balanced clustering) result in the same $\mathcal{H}$-tree. □

For the representation of a square matrix we introduce a hierarchical tree based on a product index set $\mathcal{I}\times\mathcal{I}$.

Definition 2.3.4 (Block $\mathcal{H}$-Tree $T_{\mathcal{I}\times\mathcal{I}}$) *Let $\mathcal{I}$ be a finite set and $T_{\mathcal{I}}$ the corresponding $\mathcal{H}$-tree. An $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ is called* block (cluster) $\mathcal{H}$-tree *if for all $v\in T_{\mathcal{I}\times\mathcal{I}}$ there exist $r,s\in T_{\mathcal{I}}$ such that $v = r\times s$. The vertices $v\in T_{\mathcal{I}\times\mathcal{I}}$ are called block clusters.*

Remark 2.3.5 If rectangular matrices from $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ are approximated in $\mathcal{H}$-matrix format the block tree is constructed from $T_{\mathcal{I}}$ and $T_{\mathcal{J}}$. Since throughout this work only square matrices are approximated as $\mathcal{H}$-matrices we will not consider this case. □

The set of indices is recursively subdivided as follows.

Remark 2.3.6 (Construction of $T_{\mathcal{I}\times\mathcal{I}}$) We start with the product index set

$$\operatorname{root}(T_{\mathcal{I}\times\mathcal{I}}) := \mathcal{I}\times\mathcal{I},$$

and define the set of successors according to the set of sons in $T_{\mathcal{I}}$ and some technical considerations. The product index set is hierarchically partitioned into $r\times s$ blocks, where we stop the block splitting as soon as the corresponding submatrix $M_{|_{r\times s}}$ admits a low-rank approximation. The suitable blocks are determined by a problem dependent admissibility condition. For problems based on a variational formulation of elliptic differential operators, the corresponding Green's function $G(x,y)$ has an algebraic singularity at $x=y$. Then, a standard condition is given by the following definition.

Definition 2.3.7 (Admissibility condition) *A product index set $r\times s$ is called* admissible *if the corresponding domains*

$$\tau := \bigcup_{i\in r}\operatorname{supp}(\varphi_i), \quad \sigma := \bigcup_{j\in s}\operatorname{supp}(\varphi_j)$$

fulfill the admissibility condition:

$$\min\{\operatorname{diam}(\tau),\operatorname{diam}(\sigma)\} \le 2\eta\ \operatorname{dist}(\tau,\sigma). \tag{2.27}$$

The real-valued parameter $\eta>0$ is typically chosen between $1/2$ and 5.

To limit the number of leaves in $T_{\mathcal{I}\times\mathcal{I}}$ as many as possible (preferably large) off-diagonal blocks should fulfill (2.27). With the assumption that the underlying space of ansatz functions is locally separated, i.e., there exists a moderate constant $n_{\min}$ which bounds (2.26), an effective use of the data-sparse $\mathcal{H}$-matrix format induced by $T_{\mathcal{I}\times\mathcal{I}}$ is guaranteed. In practice, the partitioning of a block $r\times s$ will be stopped as soon as a given minimum block size is reached for r or for s instead of splitting the clusters up to leaf size one. This minimum block size is chosen as large as the bound for the number of close supports $n_{\min}$ since we cannot expect to obtain admissible blocks of smaller dimension. With these notions we can define the set of successors of a vertex $r\times s\in T_{\mathcal{I}\times\mathcal{I}}$:

$$S(r\times s) := \begin{cases} \emptyset, & \text{if } r\times s \text{ fulfils (2.27)},\\ \emptyset, & \text{if } \min\{\#r,\#s\}\le n_{\min},\\ \{r'\times s' | r'\in S(r), s'\in S(s)\}, & \text{otherwise.}\end{cases}$$

□

Remark 2.3.8 (Depth of $T_{\mathcal{I}\times\mathcal{I}}$) It is shown in [77, Lemma 4.5] that under moderate assumptions on the underlying ansatz space the depth p of the block $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ is bounded by $\mathcal{O}(\log_2(n))$. This will be assumed in the following. □

Definition 2.3.9 ($\mathcal{H}$-Matrix) *Based on the constants $n_{\min}$ and k (both given in $\mathbb{N}$), we define the set of $\mathcal{H}$-*matrices *for a block $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ by*

$$\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}},k) := \left\{ M\in\mathbb{R}^{\mathcal{I}\times\mathcal{I}} \;\middle|\; \begin{array}{l} \forall r\times s\in\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}}) : \operatorname{rank}\left(M_{|_{r\times s}}\right)\le k \\ \text{or } \#r\le n_{\min} \;\text{ or } \#s\le n_{\min} \end{array} \right\}.$$

We denote by $M_{\mathcal{H}}$ the hierarchical approximation of a matrix M. In the following we omit the notation # for the cardinality of an index set whenever the meaning is clear. The set of *admissible leaves* (satisfying (2.27)) is denoted by $\mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})$. The submatrices $M_{|_{r\times s}}$ corresponding to leaves $r\times s\in\mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})$ are stored in factorized form as *Rk-matrices* (matrices of rank at most k) by (2.25). The product of two rectangular matrices has storage requirements of

$$\mathcal{S}_{\mathrm{Rk}}(M_{|_{r\times s}},k) = k(r+s)$$

entries (instead of $r\,s$). Thus, if

$$k < \min\{r,\,s\}, \tag{2.28}$$

the storage requirements of the submatrix in Rk-matrix format are reduced. The remaining *inadmissible blocks* with less than $n_{\min}$ rows or columns corresponding to leaves (which are summarized in the set $\mathcal{L}^-(T_{\mathcal{I}\times\mathcal{I}})$) are stored as usual full matrices with an amount of memory bounded by

$$\mathcal{S}_{\mathrm{full}}(M_{|_{r\times s}}) \le n_{\min}\max\{r,\,s\}.$$

By the choice of a minimum block size $n_{\min}$ we avoid the storage of very small submatrices in Rk-matrix format.

Example 2.3.10 [Example 2.3.3 ctd.] We consider the construction of an $\mathcal{H}$-matrix based on a block $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ where the basic partitioning of $\mathcal{I}$ is described in Example 2.3.3. Setting $\eta = 1.0$, the largest blocks which fulfill the admissibility condition (2.27), as illustrated in Figure 2.3, are $\{1,2\}\times\{7,8\}$ and $\{7,8\}\times\{1,2\}$ as depicted in green (light grey) in Figure 2.4 (c). These blocks have diameter $3h$ and their distance is of the same size. By a choice of $n_{\min} = 1$ we split the $\mathcal{H}$-tree up to depth 3 where some further blocks fulfill (2.27). The remaining red (dark grey) blocks belong to $\mathcal{L}^-(T_{\mathcal{I}\times\mathcal{I}})$, see Figure 2.4 (c).

□

$\mathcal{H}$-matrices can be stored with an almost linear amount of memory. The storage requirement for a matrix in the set $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}},k)$ depends on the depth p and on the sparsity of the $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ measured by the constant $\mathrm{C_{sp}}$.

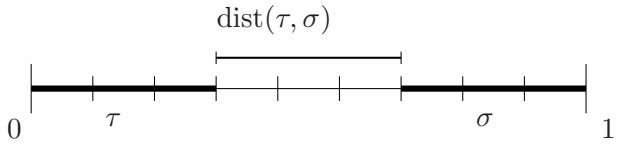
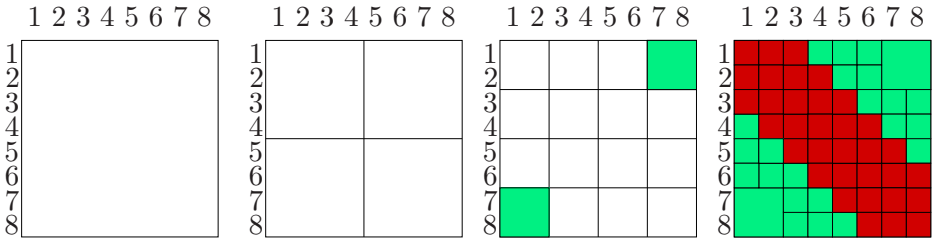


Figure 2.3: Admissibility condition in Example 2.3.10.



(a) $\mathcal{I}^{(0)} \times \mathcal{I}^{(0)}$ (b) $\mathcal{I}_1^{(1)} \times \mathcal{I}_1^{(1)}, \cdots$ (c) $\mathcal{I}_1^{(2)} \times \mathcal{I}_1^{(2)}, \cdots$ (d) $\{1\} \times \{1\}, \cdots$

Figure 2.4: Construction of an $\mathcal{H}$-matrix in Example 2.3.10.

Definition 2.3.11 (Sparsity constant $\mathrm{C_{sp}}$) *The* sparsity constant $\mathrm{C_{sp}}$ *is defined for a block $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ as follows:*

$$\mathrm{C_{sp}} := \max\left\{ \max_{r\in T_{\mathcal{I}}} \#\{s \in T_{\mathcal{I}} | r \times s \in T_{\mathcal{I}\times\mathcal{I}}\}, \max_{s\in T_{\mathcal{I}}} \#\{r \in T_{\mathcal{I}} | r \times s \in T_{\mathcal{I}\times\mathcal{I}}\} \right\}.$$

Under certain assumptions on the construction of $T_{\mathcal{I}\times\mathcal{I}}$ it is shown in [77, Lemma 4.5] that $\mathrm{C_{sp}}$ is independent of the number of indices. We assume the latter in the following.

Remark 2.3.12 ($\mathcal{H}$-matrix storage) Suppose the cardinality of the index set $\mathcal{I}$ equals n. From the definition of $\mathrm{C_{sp}}$ we derive a bound for the number of leaves in $T_{\mathcal{I}\times\mathcal{I}}$:

$$\#\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}}) \leq 2\,\mathrm{C_{sp}}\, n.$$

With a minimum block size $n_{\min}$ the required amount of real numbers for a matrix in $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ is bounded by

$$\begin{aligned}
\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) &= \sum_{r\times s\in\mathcal{L}^-(T_{\mathcal{I}\times\mathcal{I}})} \mathcal{S}_{\mathrm{full}}(T_{\mathcal{I}\times\mathcal{I}}) + \sum_{r\times s\in\mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})} \mathcal{S}_{\mathrm{Rk}}(T_{\mathcal{I}\times\mathcal{I}}, k) \\
&\leq \sum_{r\times s\in\mathcal{L}^-(T_{\mathcal{I}\times\mathcal{I}})} n_{\min} \max\{r,\, s\} + \sum_{r\times s\in\mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})} k(r+s) \\
&\leq \sum_{r\times s\in\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}})} \max\{n_{\min}, k\} r + \sum_{r\times s\in\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}})} \max\{n_{\min}, k\} s \\
&\leq 2\,\mathrm{C_{sp}}\, n(p+1) \max\{n_{\min}, k\}.
\end{aligned}$$

See [77, Lemma 2.4] for more details. For simplicity we will assume from now on that $n_{\min} \leq k$. Then, the bound for the storage requirements simplifies to

$$\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) \leq 2\,\mathrm{C_{sp}}\, n(p+1)k = \mathcal{O}(n \log_2(n) k)$$

compared to n^2 for the original (full) matrix. □

Algorithm 1 $\mathcal{T}_{\tilde{k}\leftarrow k}$: Truncated SVD for Rk-matrices.

INPUT: $R = AB^T$, $A \in \mathbb{R}^{r\times k}$, $B \in \mathbb{R}^{s\times k}$, $\tilde{k} \le k$
OUTPUT: $\mathcal{T}_{\tilde{k}\leftarrow k}(R) = \tilde{R} = \tilde{A}\tilde{B}^T$, $\tilde{A} \in \mathbb{R}^{r\times\tilde{k}}$, $\tilde{B} \in \mathbb{R}^{s\times\tilde{k}}$

1: Compute thin QR decompositions:

$$\begin{aligned} A &= Q_A R_A, \quad Q_A \in \mathbb{R}^{r\times k},\ R_A \in \mathbb{R}^{k\times k}, \\ B &= Q_B R_B, \quad Q_B \in \mathbb{R}^{s\times k},\ R_B \in \mathbb{R}^{k\times k}. \end{aligned}$$

2: Compute an SVD

$$R_A R_B^T = U\Sigma V^T.$$

3: Compute the matrix $\tilde{R} = \tilde{A}\tilde{B}^T$ of rank $\tilde{k}$ by two matrix multiplications:

$$\begin{aligned} \tilde{A} &\leftarrow Q_A U(:, 1:\tilde{k})\Sigma(1:\tilde{k}, 1:\tilde{k}), \\ \tilde{B} &\leftarrow Q_B V(:, 1:\tilde{k}). \end{aligned}$$

The approximate arithmetic is a means to close the set of $\mathcal{H}$-matrices $M \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ under addition, multiplication and inversion. These operations in formatted arithmetic are performed blockwise with exact addition or multiplication followed by truncating the resulting block back to rank k using a best Frobenius norm approximation. The corresponding truncation operator onto $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ is denoted by $\mathcal{T}_{\mathcal{H},k}$.

Definition 2.3.13 ($\mathcal{H}$-matrix truncation) *For $\mathcal{H}$-matrices $M \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$, the truncation operator*

$$\mathcal{T}_{\mathcal{H},\tilde{k}\leftarrow k} : \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) \to \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, \tilde{k}), \quad M \mapsto \tilde{M},$$

is defined blockwise for all leaves of $T_{\mathcal{I}\times\mathcal{I}}$ by

$$\tilde{M}_{|r\times s} := \begin{cases} \mathcal{T}_{\tilde{k}\leftarrow k}(M_{|r\times s}), & \text{if } r\times s \text{ admissible,} \\ M_{|r\times s}, & \text{otherwise.} \end{cases}$$

The truncation operator $\mathcal{T}_{\tilde{k}\leftarrow k}$ maps an Rk-matrix to a best Frobenius (or spectral) norm approximation of rank $\tilde{k}$ by use of a truncated singular value decomposition (SVD). Note that for $k \le \tilde{k}$ the operator $\mathcal{T}_{\tilde{k}\leftarrow k}$ is the identity.

In practice, the truncation is performed as described in Algorithm 1. We use the colon notation to designate submatrices. Adding the flops of the computational steps in Algorithm 1 as given in [74, Section 1.2.4, 5.2.9, 5.4.5], the complexity of truncating $R = AB^T$ of rank k to rank $\tilde{k}$ by a truncated SVD is given by

$$\mathcal{N}_{\mathcal{T}_{\tilde{k}\leftarrow k}}(r, s, k) = 6k^2(r+s) + 23k^3.$$

For the complexity of the $\mathcal{H}$-matrix truncation a bound is given in [77, Lemma 2.9]:

$$\mathcal{N}_{\mathcal{T}_{\mathcal{H},\tilde{k}\leftarrow k}} \le 6k\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) + 23k^3 \#\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}}).$$

By help of the truncation operator it is now possible to introduce the approximate arithmetic. For two matrices $A, B \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ and a vector $v \in \mathbb{R}^n$ we obtain approximate arithmetic operations, which all have linear-polylogarithmic complexity. More details and proofs concerning the complexity estimates in this section can be found in [77]. We consider the complexity in dependency on k. The estimates are also valid for adaptively chosen ranks where k is taken as maximum over all blockwise ranks. The adaptive choice of ranks is described at the end of this section.

Definition 2.3.14 (Matrix-vector multiplication) *The matrix-vector product for a matrix $A \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ and a vector $v \in \mathbb{R}^{\mathcal{I}}$ is defined blockwise for all leaves $r \times s \in \mathcal{L}(T_{\mathcal{I}\times\mathcal{I}})$ by Algorithm 2.*

Algorithm 2 $\mathcal{H}$-matrix-vector multiplication.

```
1: w := 0,
2: for all r × s ∈ L(T_{I×I}) do
3:     w_{|r} = w_{|r} + (A_{|r×s} v_{|s}),
4: end for
```

The matrix-vector multiplication is of the same complexity as storing an $\mathcal{H}$-matrix,

$$\mathcal{N}_{\mathcal{H}\cdot v}(T_{\mathcal{I}\times\mathcal{I}}, k) \leq 2\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) = \mathcal{O}(n \log_2(n) k),$$

see [77, Lemma 2.5].

Definition 2.3.15 ($\mathcal{H}$-matrix addition) *The formatted sum of two matrices $A, B \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ (denoted by $\oplus$) is defined by exact addition (with $A + B \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, 2k)$) followed by a truncation back into the set of $\mathcal{H}$-matrices of rank at most k:*

$$A \oplus B := \mathcal{T}_{\mathcal{H},k\leftarrow 2k}(A + B).$$

The formatted sum takes roughly

$$\begin{aligned}
\mathcal{N}_{\mathcal{H}\oplus\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) &\leq 24k\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) + 184k^3 \#\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}}) \\
&\leq 24k(2\,\mathrm{C_{sp}}\, nk(p+1)) + 184k^3 \#\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}}) \\
&= \mathcal{O}(n \log_2(n) k^2),
\end{aligned}$$

see [77, Remark 2.14]. Analogously the formatted subtraction ($\ominus$) is defined. Complexity estimates for the formatted matrix product and the approximate inverse are much more involved. Computing the approximate product of two matrices is more complex using the $\mathcal{H}$-matrix technique since the structure of the underlying $\mathcal{H}$-tree may change drastically for the (exact) matrix product. This may happen even if we consider a multiplication of two matrices which belong to the same set $A, B \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$. In the following we will point out the main properties of the formatted operations, thereby assuming a simple structure of the $\mathcal{H}$-matrix set under consideration. First of all we assume that the block $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ is idempotent (that is, the product tree $T \cdot T$ satisfies $T \cdot T = T$) or at least almost idempotent. In the latter case the distance between

$T \cdot T$ and T is measured by an idempotency constant $\mathrm{C_{id}}$. We assume that $\mathrm{C_{id}}$ is independent of the number of indices. This can be shown for geometrically balanced cluster trees, see [77, Lemma 4.5]. For more details about product trees see [77, Section 2.1.6].

Definition 2.3.16 ($\mathcal{H}$-matrix multiplication) *Let $T_{\mathcal{I}\times\mathcal{I}}$ be a block $\mathcal{H}$-tree with sparsity constant $\mathrm{C_{sp}}$ and depth p. For two matrices $A, B \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ the formatted (best) multiplication ($\odot_{best}$) is defined by the exact multiplication followed by truncation*

$$A \odot_{best} B = \mathcal{T}_{\mathcal{H},k\leftarrow\tilde{k}}(A\,B).$$

The computational complexity consists of two parts, the cost for the calculation of the exact matrix product, $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) \cdot \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) \to \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, \tilde{k})$,

$$\mathcal{N}_{\mathcal{H}\cdot\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) \leq 4{\mathrm{C_{sp}}}^3(p+1)^2k^2n,$$

and the cost for truncating large blocks of rank $\tilde{k}$ back to rank k and convert smaller leaves to full matrix format. This sums up to an overall complexity for the formatted multiplication [77, Theorem 2.24]:

$$\mathcal{N}_{\mathcal{H}\odot_{\mathrm{best}}\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) \leq 43{\mathrm{C_{id}}}^3{\mathrm{C_{sp}}}^3k^3(p+1)^3\max\{n, \#\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}})\}.$$

In practice, a truncation back to rank k is applied in each intermediate (and rank increasing) step of the matrix product. The resulting operation is called *fast multiplication* ($\odot$) and has a reduced complexity of

$$\begin{aligned}\mathcal{N}_{\mathcal{H}\odot\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) &\leq 56{\mathrm{C_{sp}}}^2\max\{\mathrm{C_{sp}}, \mathrm{C_{id}}\}k^2(p+1)^2 \\ &\quad +184\,\mathrm{C_{sp}}\,\mathrm{C_{id}}\,k^3(p+1)\#\mathcal{L}(T_{\mathcal{I}\times\mathcal{I}}) \\ &= \mathcal{O}(n\log_2^2(n)k^2).\end{aligned}$$

The fast multiplication is not guaranteed to compute best approximations to the exact matrix product since theoretically singular values can be canceled in the intermediate steps.

It is shown in [20, 22] that inverses of finite element discretizations of second-order elliptic partial differential operators with bounded coefficients can be approximated by $\mathcal{H}$-matrices with blockwise low rank. We compute approximations to the inverses via approximate LU decompositions as introduced in [51, 120] and described in the following remark.

Remark 2.3.17 ($\mathcal{H}$-LU-decomposition) A matrix $A \in \mathbb{R}^{n\times n}$ with non-zero minors is decomposed in exact arithmetic as

$$A = LU,$$

where L is a lower-triangular and U is an upper-triangular matrix. It is shown in [21, Theorem 3.4] that L and U can be approximated by $\mathcal{H}$-matrices $L_{\mathcal{H}}$ and $U_{\mathcal{H}}$ with

$$\|A - L_{\mathcal{H}}U_{\mathcal{H}}\| \leq \epsilon,$$

if each Schur complement S in A can be approximated blockwise by an $\mathcal{H}$-matrix $\tilde{S}$ with accuracy ϵ,

$$\|S_{|_{r\times s}} - \tilde{S}_{|_{r\times s}}\| \leq \epsilon, \quad \text{for each } r \times s \in \mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}}).$$

It is proven that matrices stemming from finite element discretizations of elliptic partial differential operators with measurable coefficients possess this property. For simplicity we consider the block LU-decomposition of A:

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \approx \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix}.$$

Using $\mathcal{H}$-matrix arithmetic the decomposition is recursively defined by the following four steps:

1. Compute the $\mathcal{H}$-LU-decomposition: $A_{11} = L_{11}U_{11}$;
2. Solve the triangular system $A_{12} = L_{11}U_{12}$ for U_{12};
3. Solve the triangular system $A_{21} = L_{21}U_{11}$ for L_{21};
4. Compute the $\mathcal{H}$-LU-decomposition: $A_{22} \ominus L_{21} \odot U_{12} = L_{22}U_{22}$.

The triangular solves in 2 and 3 are recursively defined via block forward and backward substitutions based on formatted arithmetic. The recursive definitions of the $\mathcal{H}$-LU-decompositions in step 1 and 4 lead to the computation of standard LU-decompositions on the coarsest level, that is in the inadmissible blocks on the diagonal. The storage requirements for the two factors $L_{\mathcal{H}}$ and $U_{\mathcal{H}}$ are of the same size as those for the matrix $A_{\mathcal{H}}$:

$$\mathcal{S}_{\mathcal{H}-LU}(T_{\mathcal{I}\times\mathcal{I}}, k) = \mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) = \mathcal{O}(n \log_2(n) k).$$

The complexity of the $\mathcal{H}$-LU-decomposition can be estimated by the complexities of the above mentioned computational steps. It is shown in [80, Corollary 4] that the computational work breaks down to the complexity of the formatted matrix multiplication and is therefore bounded if the block tree $T_{\mathcal{I}\times\mathcal{I}}$ is constructed by geometric bisection by

$$\mathcal{N}_{\mathcal{H}-LU}(T_{\mathcal{I}\times\mathcal{I}}, k) \leq \mathcal{N}_{\mathcal{H}\odot\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k) = \mathcal{O}(n \log_2^2(n) k^2).$$

If additionally the cardinality of the clusters is balanced, it is moreover shown that

$$\mathcal{N}_{\mathcal{H}-LU}(T_{\mathcal{I}\times\mathcal{I}}, k) \approx \frac{1}{2}\mathcal{N}_{\mathcal{H}\odot\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, k).$$

□

The $\mathcal{H}$-LU-decomposition is used to compute an approximate inverse $A_{\mathcal{H}}^{-1}$ of A as described in Algorithm 3. The computational complexity for the triangular solves can be bounded by half the complexity of the formatted matrix product. Therefore, the cost for computing the approximate inverse is bounded by the cost for computing the product of two $\mathcal{H}$-matrices and thus is in the size of

Algorithm 3 Calculate approximate inverse $A_{\mathcal{H}}^{-1}$.

1: Decompose $A_{\mathcal{H}}$ by $\mathcal{H}$- LU-decomposition:

$$[L_{\mathcal{H}}, U_{\mathcal{H}}] \leftarrow LU_{\mathcal{H}}(A_{\mathcal{H}}).$$

2: Compute Y by $\mathcal{H}$-based forward substitution

$$L_{\mathcal{H}} Y = (I_n)_{\mathcal{H}}.$$

3: Compute X by $\mathcal{H}$-based backward substitution

$$U_{\mathcal{H}} X = Y.$$

4: Define the approximate inverse $A_{\mathcal{H}}^{-1} := X$.

$\mathcal{N}_{\mathcal{H}-LU}(T_{\mathcal{I}\times\mathcal{I}}, k)$. Note that the approximate inversion $A_{\mathcal{H}}^{-1}$ can also be computed by using the Frobenius formula (obtained by block Gaussian elimination on A under the assumption that all principal submatrices of A are nonsingular) with formatted addition and multiplication. This approach has larger storage requirements since it takes roughly three times the workspace occupied by the original matrix.

Adaptive Arithmetic. Instead of fixing the rank for each block we prespecify the relative accuracy in each matrix block $M_{|_{r\times s}}$ by a parameter $\epsilon \in (0,1)$. The corresponding rank will be determined adaptively

$$k(\epsilon, M_{|_{r\times s}}) := \min\{k \in \mathbb{N} |\, \sigma_{k+1} \leq \epsilon\, \sigma_1\}, \tag{2.29}$$

assuming that the singular values σ_i, $i = 1, \ldots, \min\{r, s\}$ of $M_{|_{r\times s}}$ are in descending order. Then, the storage requirements for a matrix in $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, \epsilon)$ can be bounded as for matrices in $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ if k is taken as maximum over all blockwise ranks. In discretized control problems for PDEs as in the investigated examples, defined on $\Omega \subset \mathbb{R}^d$, it is observed that $k \sim \log^{d-1}(1/\epsilon)$ is sufficient to obtain a relative approximation error of $\mathcal{O}(\epsilon)$. It is proven in [44, Theorem 8] that the blockwise ranks $k \sim \log^{d+1}(1/\epsilon)$ are adequate. Earlier results can be found in [22]. Therefore, the storage requirements for a matrix in $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, \epsilon)$ are bounded by

$$\mathcal{S}_{\mathcal{H}}(T_{\mathcal{I}\times\mathcal{I}}, \epsilon) \leq 2\, \mathrm{C_{sp}}\, n(p+1) \log^{d+1}(1/\epsilon) = \mathcal{O}(n \log_2(n) \log^{d+1}(1/\epsilon)),$$

assuming that the relation $k = \mathcal{O}(\log^{d+1}(1/\epsilon))$ holds.

The truncation operator $\mathcal{T}_{\mathcal{H}}$ in the definition of the approximate arithmetic has to be adapted such that the approximate matrix operations are exact up to ϵ in each matrix block. This is done by replacing the truncation operator $\mathcal{T}_{\tilde{k}\leftarrow k}$ in Definition 2.3.13 by $\mathcal{T}_{\epsilon}$. The operator $\mathcal{T}_{\epsilon}$ maps a Rk-matrix R to a matrix of rank $k(\epsilon, R)$ which is determined by (2.29). The truncation to rank $k(\epsilon)$ is performed in the same way as described in Algorithm 1 for the operator $\mathcal{T}_{\tilde{k}\leftarrow k}$.

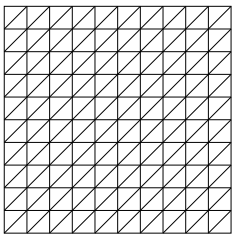


Figure 2.5: Uniform grid.

The error between R and the truncated matrix $\mathcal{T}_\epsilon(R)$ is bounded by

$$\frac{\|\mathcal{T}_\epsilon(R) - R\|_2}{\|R\|_2} \leq \epsilon.$$

Using the truncation operator $\mathcal{T}_{\mathcal{H},\epsilon}$ in the formatted arithmetic defines the so-called *adaptive arithmetic*.

At the end of this section we illustrate the use of the hierarchical matrix format for a typical application.

Example 2.3.18 We consider a stable symmetric matrix

$$A := -L^{-1}\hat{A}L^{-T}, \tag{2.30}$$

where L is a Cholesky factor of the symmetric, positive definite mass matrix E and $\hat{A}$ is the stiffness matrix, resulting from a FEM discretization of the two-dimensional heat equation in the unit square $\Omega = [0,1]^2$. The matrices $E \in \mathbb{R}^{n\times n}$, $\hat{A} \in \mathbb{R}^{n\times n}$ are given by the entries

$$\begin{array}{rcl} E_{ij} & = & \int_\Omega \varphi_i(\xi)\varphi_j(\xi)\,d\xi, \\ \hat{A}_{ij} & = & \int_\Omega \nabla\varphi_i(\xi)\cdot\nabla\varphi_j(\xi)\,d\xi, \quad \text{for } i,j = 1,\ldots,n. \end{array} \tag{2.31}$$

We use a uniform triangulation of the domain as depicted in Figure 2.5 with n inner grid points. The uniform grid is known to be a bad triangulation with respect to complexity of the formatted arithmetic [77]. It will be used in the following because of its simplicity and the results can be regarded as worst case performances. With a given minimum block size $n_{\min} := 256$, using the standard admissibility condition (2.27) with $\eta := 1.0$, and the geometrically balanced clustering, we approximate A as $\mathcal{H}$-matrix with different values for the blockwise accuracy ϵ. For the approximation of mass and stiffness matrices and also of their inverses it is advised to use a special $\mathcal{H}$-matrix structure where all off-diagonal blocks are approximated as Rk-matrices. For the problem size $n = 4096$ and $\epsilon = 10^{-4}$, the $\mathcal{H}$-matrix with a blockwise SVD of all Rk-submatrices and the blockwise ranks is shown in Figure 2.6. The singular values of each admissible block are depicted in logarithmic scale from 10^{-15} to 10^0. The inadmissible blocks at the diagonal are displayed in red (dark grey).

For different problem sizes, the maximum rank k over all admissible blocks, the resulting values for the constants C_{sp} and C_{id}, and the storage requirements

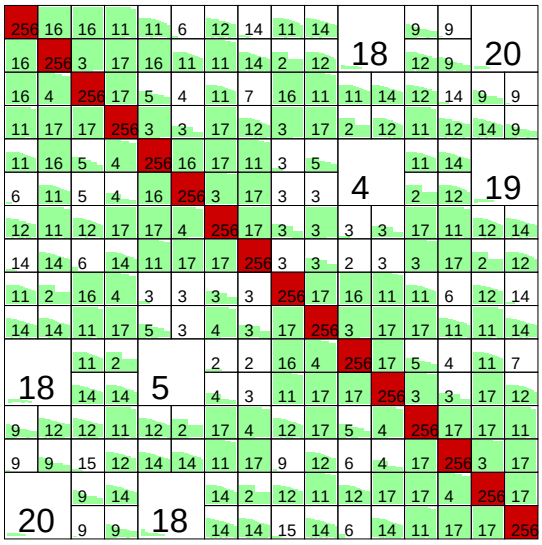


Figure 2.6: Blockwise SVD of $A_{\mathcal{H}}$ with $n = 4096$, $\epsilon = 10^{-4}$.

n	ϵ	depth	C_{sp}	C_{id}	k	$\mathcal{S}$ (MB)	$\frac{\|A-A_{\mathcal{H}}\|_2}{\|A\|_2}$
1024	1.e-04	3	4	1	17	2.64	2e-05
1024	1.e-06	3	4	1	18	2.69	8e-08
1024	1.e-08	3	4	1	21	2.76	3e-10
1024	1.e-16	3	4	1	31	2.96	2e-15
4096	1.e-04	5	16	5	20	17.53	2e-05
4096	1.e-06	5	16	5	27	19.86	8e-08
4096	1.e-08	5	16	5	32	21.99	4e-10
4096	1.e-16	5	16	5	47	27.68	2e-15
16,384	1.e-04	7	30	17	22	109.79	2e-05
16,384	1.e-06	7	30	17	28	130.52	8e-08
16,384	1.e-08	7	30	17	32	151.15	4e-10
16,384	1.e-16	7	30	17	53	204.95	4e-15
65,536	1.e-04	9	30	17	22	538.36	-
262,144	1.e-04	11	30	17	22	2372.79	-

Table 2.1: Some accuracy and complexity results for $A_{\mathcal{H}}$ in Example 2.3.18.

for $A_{\mathcal{H}}$ denoted by $\mathcal{S}$ in megabyte (MB) are summarized in Table 2.1. Furthermore, the relative errors between $A_{\mathcal{H}}$ and A for n up to $16,384$ are given. Note that for the full matrices the storage requirements increase from 8 MB for $n = 1024$, over 2048 MB for $n = 16,384$, up to 512 gigabyte (GB) storing $A \in \mathbb{R}^{n\times n}$ with $n = 262,144$. The storage requirements for $A_{\mathcal{H}}$ are reduced significantly compared to the amount needed for the full matrix $A \in \mathbb{R}^{n\times n}$. □

Chapter 3

Linear Matrix Equations

This chapter is concerned with the numerical solution of linear matrix equations. For given matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $W \in \mathbb{R}^{n\times m}$ we seek a solution $X \in \mathbb{R}^{n\times m}$ of a *Sylvester equation*,

$$AX + XB + W = 0, \tag{3.1}$$

its symmetric variant, a *Lyapunov equation*,

$$AX + XA^T + W = 0, \quad W = W^T, \tag{3.2}$$

and of a *Stein equation*, also called discrete Lyapunov equation,

$$AXA^T - X + W = 0. \tag{3.3}$$

Over the last 40 years, numerous numerical methods have been introduced to solve these linear matrix equations. They can be divided into direct and iterative methods. Direct approaches first transform the coefficient matrices A and B to obtain a system with a structure that admits a simple solution. The dense Bartels-Stewart algorithm, developed in 1972 [14] for Sylvester and Lyapunov equations, first reduces the matrices A, B to real Schur form by the QR algorithm. For Sylvester equations, a more efficient method independently derived in [61] and [73] shows that it is sufficient to reduce A to Hessenberg form. The resulting linear systems are solved by a backsubstitution process. For the discrete Stein equation (3.3), direct methods can be found for instance in [13, 73, 149]. The direct approaches can be generalized to so that they solve generalized matrix equations, see e.g. [70]. The algorithms are implemented in the MATLAB Control System Toolbox as functions `lyap` and `dlyap` and in the SLICOT Basic Control Toolbox [27]. All these methods are backward stable and have cubic complexity. The storage requirements are of order $\mathcal{O}(n^2)$. Hence they are not appropriate for large-scale computations. However, using the recent MATLAB distribution R2007a [125], larger Lyapunov equations with order up to $n = O(10^5)$ can be solved with direct methods provided that sufficient memory is available.

Apart from direct methods, there are several iterative methods, for example the Smith method [152], the alternating direction implicit (ADI) iteration method [123, 166], and the Smith(l) method [134, 135]. These methods

are particularly suitable for large-scale sparse matrix equations but compute the solution in dense form with a quadratic amount of storage. There are many approaches to solve large-scale Sylvester and Lyapunov equations using Krylov subspace methods [99, 101], but in general they are inferior to ADI and Smith-type methods, see [134]. Furthermore, several multigrid methods are proposed [78, 132, 145].

In the context of model reduction, those methods are favorable which take a special structure of the matrix equation into account. That is, the "right-hand side term" is given in factorized form, see for instance a special Sylvester equation (for the computation of the cross-Gramian of an LTI system),

$$AX + XA + FG = 0, \tag{3.4}$$

a (standard) Lyapunov equation,

$$AX + XA^T + BB^T = 0, \tag{3.5}$$

a generalized Lyapunov equation,

$$AXE^T + EXA^T + BB^T = 0, \tag{3.6}$$

and a Stein equation,

$$AXA^T - X + BB^T = 0, \tag{3.7}$$

where $X \in \mathbb{R}^{n\times n}$, F, G^T, $B \in \mathbb{R}^{n\times m}$ and $A \in \mathbb{R}^{n\times n}$ is stable. For generalized equations it is assumed that $E \in \mathbb{R}^{n\times n}$ is nonsingular and that the matrix pencil $\lambda E - A$ is regular and stable. All these matrix equations are related to LTI systems where typically the constant terms FG and BB^T are of low rank $m \ll n$. In most cases it is often observed that under these assumptions the solution of a matrix equation has low rank or at least low numerical rank, with fast eigenvalue decay rate, see, e.g., [136, 4, 157, 76]. Then, the memory requirements can be considerably reduced by computing the solution in factorized form. For the symmetric Lyapunov or Stein equation the solution X is symmetric positive semi-definite, such that it can be factorized into $X = YY^T$. The factor Y can be given as

- *Cholesky factor* of X, i.e., $Y \in \mathbb{R}^{n\times n}$ is a square lower triangular matrix, or as
- a *full-rank factor* of X, i.e., $Y \in \mathbb{R}^{n\times \operatorname{rank}(X)}$ is a rectangular matrix.

Furthermore, in many applications as, for instance, in the SR method for balanced truncation [115, 161], see also Section 5.1, only the Cholesky factors are required instead of the solution X. An important advantage of this approach is that the condition number of the Cholesky factor is the square root of the condition number of X. Therefore, we expect higher accuracy in subsequent computations due to better conditioning of Y.

The direct computation of Cholesky factors of X for the solution of Lyapunov (3.5) and Stein equations (3.7) via the Bartels-Stewart approach is suggested by Hammarling in [92, 93], see also [164], and revised by Kressner in

[109]. It is implemented in the SLICOT-based MATLAB functions `lyapchol` and `dlyapchol`. Penzl in [133] extended the method to generalized Lyapunov equations (3.6). Since these techniques still require $\mathcal{O}(n^3)$ flops and $\mathcal{O}(n^2)$ memory, they are only practicable for problems of relatively small size.

There are many iterative methods which exploit the low-rank property particularly for the solution of large-scale, sparse Lyapunov equations, see, e.g., methods based on a low-rank ADI or Smith iterations as the Cholesky factor ADI algorithm [117, 119, 135], cyclic low-rank Smith methods [86, 135] and parallelizations of the low-rank ADI iteration [5], and projection-type methods [100, 101, 102, 146, 104]. The latter class is known for computing solutions of lower accuracy or, by performing much more iteration steps, solutions of relatively large numerical ranks [135]. A special case in this class, proposed in [150], appears to be quite competitive to ADI and Smith methods. Most of these methods are particularly adapted and efficient for the solution of large-scale sparse problems. However, all iterative methods are still of cubic complexity when applied to dense problems.

Here, we will focus on the sign function method, introduced first in 1971 by Roberts [144]. Roberts and also Beavers and Denman [19] used the matrix sign function for solving algebraic Riccati, Sylvester, and Lyapunov equations. A related iterative scheme in discrete-time is the squared Smith iteration. These methods have certain attractive properties as they can be parallelized easily and usually show fast convergence [26, 35]. Throughout this work they are the methods of choice since they are well adapted for the purpose of model reduction and allow for the use of approximate arithmetic which results in very efficient data-sparse solvers for large-scale matrix equations as described in Chapter 4. In the sign function iteration the special structure of the "right-hand side" can be exploited for the direct computation of Cholesky factors [114]. In 1999 Benner and E.S. Quintana-Ortí extended the idea to stable generalized Lyapunov equations [31]. Furthermore, using column compression in each iteration step, it is shown that the algorithm converges to an (approximate) full-rank factor of a low-rank approximation to the solution X. The number of columns of the so-called *approximate low-rank factor* are limited by the numerical rank of X such that the storage requirements for the solution are reduced. An efficient algorithm for the solution of coupled Lyapunov equations, as required in model reduction methods with system-theoretic background, is presented in [25]. The technique is also applicable for the numerical solution of large-scale Sylvester equations with factorized "right-hand side" (3.4) [23, 38]. The squared Smith method can be modified to compute an approximate low-rank factor of the solution X from the Stein equation (3.7) directly [35].

In this chapter we review some properties of the sign function iteration and of the squared Smith iteration for the different types of matrix equations. The modified approaches for computing approximate low-rank solution factors for Sylvester equations (3.4) (in general form), (standard) Lyapunov equations (3.5), generalized Lyapunov equations (3.6) and Stein equations (3.7) are described in the corresponding sections.

3.1 Sylvester Equations

This section is concerned with the numerical solution of linear matrix equations of the following form:

$$AX + XB + W = 0, \tag{3.8}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $W \in \mathbb{R}^{n\times m}$ and a matrix X of $n \times m$ unknowns. Matrix equations of this type are called *Sylvester equations*.

The Sylvester equation is an important tool in numerical linear algebra, e.g. for model reduction using the cross-Gramian approach, in observer design, where a state observer is constructed from the solution of a Sylvester equation and for the purpose of matrix block diagonalization, i.e.,

$$\begin{bmatrix} I & -X \\ 0 & I \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} I & X \\ 0 & I \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}$$

with X solving the Sylvester equation $A_{11}X - XA_{22} + A_{12} = 0$.

We get an equivalent representation of (3.8) by using the Kronecker product and the vec-operator

$$(I_m \otimes A + B^T \otimes I_n)\,\vec{X} = -\vec{W}. \tag{3.9}$$

This vectorized representation immediately leads to solvability conditions. Since

$$\Lambda(I_m \otimes A + B^T \otimes I_n) = \{\lambda_i + \mu_j |\, i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, m\},$$

with $\lambda_i \in \Lambda(A)$ and $\mu_j \in \Lambda(B)$, the system matrix in (3.9) is nonsingular, if and only if, the spectra of A and $-B$ are disjoint. This, in turn, is equivalent to the existence and uniqueness of the solution X of (3.8) [113]. Furthermore, if all eigenvalues of A and B have negative real part, we obtain an explicit solution formula [111]:

$$X = \int_0^\infty e^{At}\, W\, e^{Bt} dt.$$

Sylvester equations which have this property are called *stable Sylvester equations*. In the following, we generally assume stability of the Sylvester equation under consideration.

For the numerical solution of Sylvester equations we first consider the direct methods as they are implemented in the MATLAB Control System Toolbox and in the SLICOT Basic Control Toolbox. The Bartels-Stewart algorithm [14] reduces the coefficient matrices A and B to real Schur form by the QR algorithm. The solution matrix X is then computed via a backsubstitution process involving the solution of recursively constructed triangular systems of equations. In the Hessenberg-Schur method [61, 73], A is transformed to upper Hessenberg form, B to real upper Schur form, and the solution of the transformed system is computed by backsubstitution. The overall complexity assuming that the Schur decomposition of a $n \times n$ matrix requires $25n^3$ flops [74, Section 7.5.6] is given by

$$\mathcal{N}_{\mathrm{BS}}(n, m) = 25n^3 + 25m^3 + 5mn(m + n)$$

operations for the Bartels-Stewart algorithm [74, Section 7.6.3]. The Hessenberg-Schur method reduces the complexity for transforming A to upper Hessenberg form to $\frac{14}{3}n^3$ flops [74, Section 7.4.3]. Note that it is possible to improve the performance solving the triangular systems by recursive blocked algorithms [105]. The Bartels-Stewart method requires at least $2n^2 + 2m^2 + nm$ storage whereas the demand is slightly increased by Hessenberg-Schur given by $3n^2 + 2m^2 + nm$.

Instead of direct methods we consider iterative schemes for the solution of matrix equations. We will focus on the sign function method which was originally developed by Roberts in 1971 [144] to solve algebraic Riccati equations. We first describe some basic properties of the matrix sign function, then their impact on the numerical solution of matrix equations.

3.1.1 The Sign Function Iteration

Consider a square matrix $Z \in \mathbb{R}^{n\times n}$ with no eigenvalues on the imaginary axis. By the real variant of the Jordan canonical form there exists a nonsingular matrix $S \in \mathbb{R}^{n\times n}$ such that

$$Z = S^{-1} \begin{bmatrix} J_\ell^+ & 0 \\ 0 & J_{n-\ell}^- \end{bmatrix} S, \tag{3.10}$$

where the upper block corresponds to the eigenvalues of Z with positive real part and $J_{n-\ell}^-$ contains the Jordan blocks belonging to the other eigenvalues, i.e., $\Lambda(J_\ell^+) \subset \mathbb{C}^+$, $\Lambda(J_{n-\ell}^-) \subset \mathbb{C}^-$.

Definition 3.1.1 (Matrix sign function [19]) *The* matrix sign function *for $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) \cap \jmath\mathbb{R} = \emptyset$ is defined as*

$$sign(Z) := S^{-1} \begin{bmatrix} I_\ell & 0 \\ 0 & -I_{n-\ell} \end{bmatrix} S$$

with S given by (3.10).

The matrix sign function for square matrices was introduced by Roberts [144] in 1971. He defined it via a contour integral, for other equivalent definitions see [107]. The matrix sign function has many applications in systems theory and matrix computations as it is used to compute eigenvalues and invariant subspaces and can be applied to the solution of Riccati and continuous linear matrix equations. We review some basic properties of the matrix sign function which will be used throughout this section.

Lemma 3.1.2 *The matrix sign function for a matrix $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) \cap \jmath\mathbb{R} = \emptyset$ has the following properties:*

a) $(sign(Z))^2 = I_n$,

b) $sign\left(T^{-1}ZT\right) = T^{-1}sign(Z)\,T$, *for all nonsingular* $T \in \mathbb{R}^{n\times n}$,

c) $sign(Z) = -I_n$ *if Z is Hurwitz,*

(d) $P_+ := (I_n + sign(Z))/2$ *is the spectral projector onto the eigenspace corresponding to* $\Lambda(Z) \cap \mathbb{C}^+$,

(e) $P_- := (I_n - sign(Z))/2$ *is the spectral projector onto the eigenspace corresponding to* $\Lambda(Z) \cap \mathbb{C}^-$.

It follows from Lemma 3.1.2 a) that we may calculate the sign of a matrix Z by applying a Newton iteration to the solution of $Z^2 - I_n = 0$:

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{j+1} &\leftarrow \frac{1}{2}(Z_j + Z_j^{-1}), \quad j = 0, 1, 2, \ldots. \end{aligned}$$

The iteration is well defined that means Z_{j+1} is invertible if Z_j is invertible, since $\Lambda(Z_j) \cap \jmath\mathbb{R} = \emptyset$ yields $\Lambda(Z_j + Z_j^{-1}) \cap \jmath\mathbb{R} = \emptyset$. The so-called *sign function iteration* converges globally and locally quadratically in a neighborhood of $\operatorname{sign}(Z)$ [107],

$$\lim_{j\to\infty} Z_j = \operatorname{sign}(Z),$$

and is well behaved in finite-precision arithmetic [53]. There are many other iterative schemes proposed for the solution of the matrix sign function, for an overview see [107], but this one seems to provide the most robust and fastest implementation and is suitable for parallel computation [38].

The matrix sign function can be sensitive to perturbations if the departure of Z from normality is high or if the stable and unstable eigenvalues are not well separated [74, Section 7.2.4]. This result can be found in a detailed perturbation analysis in [53]. Since the matrix sign function is undefined for matrices with purely imaginary eigenvalues, the convergence as well as the accuracy of the Newton iteration may be significantly disturbed for matrices with eigenvalues close to the imaginary axis. In most of the control applications considered in the present work, the arising matrices are normal and the eigenvalues are further than $\mathrm{EPS}^{1/2}$ apart from the imaginary axis. Then, the sign function avoids the problem of separating possibly clustered eigenvalues which might cause problems or ill-conditioning when using the Bartels-Stewart algorithm. A discussion in [8, 9] suggests that if we want to compute the sign function of a matrix Z up to an accuracy of $\mathrm{EPS}^{1/2}$, the condition number of Z should be smaller than $\mathrm{EPS}^{-1/2}$.

Despite the quadratic convergence rate, slow convergence is observed if the norm of the inverse $\|Z_j^{-1}\|$ is small compared to $\|Z_j\|$. Then the iterates Z_{j+1} and Z_j are approximately equal. This is typically the case during initial iteration steps and it is therefore advised to scale the iterates to accelerate the initial convergence. Note that it is also possible to improve the numerical stability by scaling the Newton iteration. For a typical convergence history of an unscaled iteration see the solid line ("$\diamond$") in Figure 3.1.

Remark 3.1.3 (Scaling strategies) There are several scaling strategies proposed for the matrix sign function, see, e.g., in [8, 106]. Since $\operatorname{sign}(Z) =$

sign (cZ) for any parameter $c > 0$, we can incorporate scaling in the sign function iteration as follows:

$$Z_{j+1} \leftarrow \frac{1}{2}(c_j Z_j + \frac{1}{c_j} Z_j^{-1}), \tag{3.11}$$

with properly chosen scaling factors $c_j > 0$. The speed of convergence is determined by the convergence of the eigenvalues of Z to ± 1. An optimal choice for the scaling parameter is presented in [106, Theo. 3.6]. This parameter is very hard to implement since all eigenvalues of the Z-iterates have to be known. Therefore, we restrict our attention to suboptimal techniques which yield nearly optimal performance and allow for an efficient implementation.

Two common choices for scaling parameters are the *spectral scaling* given by

$$c_j^{(\text{spect.})} = \sqrt{\rho(Z_j^{-1})/\rho(Z_j)},$$

and the *determinantal scaling* suggested by Byers [52] defined by

$$c_j^{(\text{det.})} = \frac{1}{|\det(Z_j)|^{1/n}}.$$

The spectral scaling is asymptotically optimal but might cause slower convergence for problems with eigenvalues close to the imaginary axis [106, Theo. 3.7], whereas the determinantal scaling performs also well for these critical eigenvalues but is not asymptotically optimal. Determinantal scaling factors can simply be derived from triangular factors of Z_j as already computed during the inversion of Z_j.

Motivated by the optimal choice of scaling parameters for a Newton iteration to compute the polar decomposition of a matrix Z, which is given by

$$c_j^{(\text{opt.})} = \sqrt{\frac{\|Z_j^{-1}\|_2}{\|Z_j\|_2}}, \tag{3.12}$$

also a good acceleration of the initial convergence for the matrix sign iteration is observed [106], approximating the spectral norm in (3.12) by use of the power method. We will denote this scaling parameter in the following by $c^{(\text{opt.})}$ even though it is not optimal in the context of the matrix sign function. Note that $c^{(\text{opt.})}$ and $c^{(\text{spect.})}$ coincide for normal matrices. It is possible to avoid the demanding computation of the spectral norm by using the Frobenius norm

$$c_j^{(\text{Frob.})} = \sqrt{\|Z_j^{-1}\|_F/\|Z_j\|_F},$$

or the geometric mean of the 1-norm and the maximum norm

$$\|\cdot\|_2 \approx \sqrt{\|\cdot\|_1 \|\cdot\|_\infty},$$

as proposed by Higham [96]. The resulting technique is called *approximate norm scaling* with scaling parameter given by

$$c_j^{(\text{approx.})} = \sqrt{\|Z_j^{-1}\|_1 \|Z_j^{-1}\|_\infty / \|Z_j\|_1 \|Z_j\|_\infty}.$$

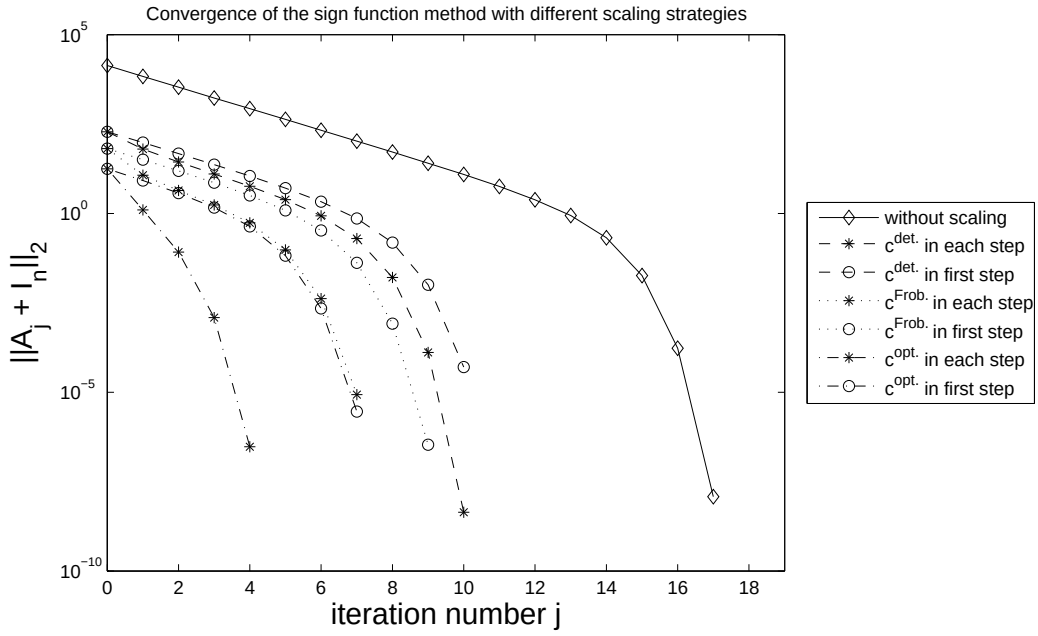


Figure 3.1: Convergence history for sign(A) with $n = 1024$.

The optimal norm scaling and related choices are used to balance the spectral norms of the summands in (3.11) such that $\|Z_j^{-1}\|_2$ and $\|Z_j\|_2$ are not "too" different. This has a beneficial effect on the numerical stability of the sign function iteration.

The influence of scaling on the speed of convergence, especially during the initial steps of the Newton iteration, is shown for some of the scaling strategies in Figure 3.1. Here, the number j of the iterations is plotted against the computed values $\|A_j + I_n\|_2$ for the stopping criterion:

$$\|A_j + I_n\|_2 \leq 10\, n\, \sqrt{\text{EPS}}.$$

The scaled sign function iteration (3.11) is applied to a stable symmetric matrix $A \in \mathbb{R}^{n\times n}$ which stems from a FEM discretization of the two-dimensional heat equation as introduced in (2.30) for the Example 2.3.18. It is observed that we obtain fastest convergence if $c^{(\text{opt.})}$ is applied in each iteration step. The performance is also examined if A_j is scaled only in the first step of the sign function iteration. In this case a few more iteration steps are necessary to reach the stopping criterion for each choice of scaling parameter. Using the Frobenius norm once instead of the spectral norm, results in some additional iterations whereas applying this scaling in each step performs comparably well to the "optimal" norm scaling used in the first step only. The determinantal scaling performs worse but is clearly better than using no scaling at all. □

We can use a special decoupling property of X in conjunction with the matrix sign function applied to a special matrix Z, to compute the solution X of a

Sylvester equation (3.8) with A and B being Hurwitz. To see this consider the following block upper triangular matrix Z defined by the coefficients of the matrix equation (3.8)

$$Z = \begin{bmatrix} A & W \\ 0 & -B \end{bmatrix}$$

and a similarity transformation

$$T = \begin{bmatrix} I_n & X \\ 0 & I_m \end{bmatrix}, \tag{3.13}$$

where in the upper right block X denotes the solution of (3.8). Then Z is block diagonalized by T:

$$\begin{aligned} T^{-1}ZT &= \begin{bmatrix} I_n & X \\ 0 & I_m \end{bmatrix}^{-1} \begin{bmatrix} A & W \\ 0 & -B \end{bmatrix} \begin{bmatrix} I_n & X \\ 0 & I_m \end{bmatrix} = \begin{bmatrix} A & AX + XB + W \\ 0 & -B \end{bmatrix} \\ &= \begin{bmatrix} A & 0 \\ 0 & -B \end{bmatrix}. \end{aligned}$$

Corollary 3.1.4 *Consider a matrix $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) \cap \jmath\mathbb{R} = \emptyset$ which is block diagonalized by a similarity transformation T as given in (3.13). The matrix sign function gives an expression for the solution X of the Sylvester equation (3.8) using Lemma 3.1.2 b) and c),*

$$sign(Z) = T\, sign\left(\begin{bmatrix} A & 0 \\ 0 & -B \end{bmatrix}\right) T^{-1} = T \begin{bmatrix} -I_n & 0 \\ 0 & I_m \end{bmatrix} T^{-1} = \begin{bmatrix} -I_n & 2X \\ 0 & I_m \end{bmatrix}.$$

By applying the sign function iteration to Z, an iterative scheme for computing the solution of the Sylvester equation (3.8) is obtained:

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{j+1} &\leftarrow \frac{1}{2}(Z_j + Z_j^{-1}) \\ &= \begin{bmatrix} \frac{1}{2}(A_j + A_j^{-1}) & \frac{1}{2}(W_j + A_j^{-1}W_jB_j^{-1}) \\ 0 & -\frac{1}{2}(B_j + B_j^{-1}) \end{bmatrix}, \quad j = 0, 1, 2, \ldots. \end{aligned}$$

The solution X of (3.8) can simply be found in the upper right block of the limit,

$$\operatorname{sign}(Z) = \lim_{j\to\infty} Z_j = \begin{bmatrix} -I_n & 2X \\ 0 & I_m \end{bmatrix},$$

as described in [144]. In this paper it was also observed that the iteration can be decoupled into three parts. Setting $A_0 \leftarrow A$, $B_0 \leftarrow B$ and $W_0 \leftarrow W$, the Newton iteration can be rewritten as

$$\begin{aligned} A_{j+1} &\leftarrow \frac{1}{2}(A_j + A_j^{-1}), \\ B_{j+1} &\leftarrow \frac{1}{2}(B_j + B_j^{-1}), \\ W_{j+1} &\leftarrow \frac{1}{2}(W_j + A_j^{-1}W_jB_j^{-1}), \quad j = 0, 1, 2, \ldots \end{aligned} \tag{3.14}$$

for computing the solution

$$X = \frac{1}{2} \lim_{j \to \infty} W_j$$

of the Sylvester equation. Since the matrices A and B are supposed to be stable, we can apply Lemma 3.1.2 c) to obtain a simple stopping criterion. From $\lim_{j\to\infty} A_j = -I_n$ and $\lim_{j\to\infty} B_j = -I_m$, we decide to stop the iteration if the A_j- and B_j-iterates are close to the identity:

$$\max\{\|A_j + I_n\|, \|B_j + I_m\|\} \leq \text{tol}, \tag{3.15}$$

with a user-defined tolerance tol. By an appropriate choice of norm and tolerance and by performing two additional iteration steps as proposed in [30], the required accuracy is reached in general owing to the quadratic convergence of the Newton iteration. Often it is proposed to choose the tolerance as the machine precision times a problem dependent constant. In our algorithms derived in the next chapter it will be preferable to set the tolerance to the size of the introduced approximation errors in order to avoid stagnation.

Complexity. Implementations of the sign function iteration typically require workspace for the matrices A, B and W and additional space of the same size. This adds to an amount of $2(n^2 + m^2 + nm)$ real numbers for the sign function iteration. The computational complexity of the iteration scheme is basically determined by the inversion of the $n \times n$ and $m \times m$ matrices in the upper two lines of (3.14) which adds to $2n^3$ and $2m^3$ flops, respectively. We assume that the factors from LU factorizations used for computing A_j^{-1} and B_j^{-1} are available. Then, during the W_j-iteration we only have to solve two linear systems via triangular systems. That additionally requires $2nm(n + m)$ flops for computing the solution of the Sylvester equation. Taking all flops together results in an overall complexity of

$$\mathcal{N}_{Sign}(n, m) = 2n^3 + 2m^3 + 2nm(n + m)$$

for one iteration step of (3.14). Taking instead of the traditional matrix inversion by Gaussian elimination a Gauss-Jordan elimination results in the same number of flops. Note that numerical experiments demonstrate that often about 10 iterations are needed for convergence [38].

Next, we consider a modification of the iteration scheme, which results in reduced computational complexity.

3.1.2 Factorized Solution of Sylvester Equations

In this section we consider a special variant of the iteration scheme (3.14) to compute the solution X of

$$AX + XB + FG = 0, \tag{3.16}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$ stable and $F \in \mathbb{R}^{n\times p}$, $G \in \mathbb{R}^{p\times m}$, in factorized form as $X = YZ$.

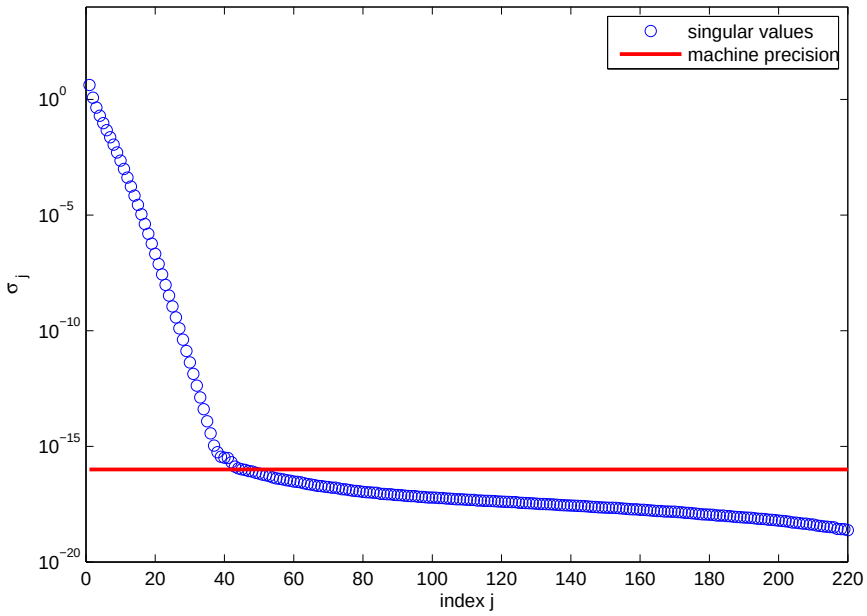


Figure 3.2: Singular values of X in (3.16) with problem sizes $n = 250$ and $m = 230$.

This method is of special interest in large-scale computations if the solution X has low rank, $\operatorname{rank}(X) \ll n, m$, or at least low numerical rank. In the first case we obtain full-rank factors $Y \in \mathbb{R}^{n \times \operatorname{rank}(X)}$, $Z \in \mathbb{R}^{\operatorname{rank}(X) \times m}$ of X. The latter case is of particular relevance; in many large-scale applications it can be observed that the eigenvalues of X decay rapidly. In [76] it is shown that the singular values of X decay exponentially if the "right-hand side" FG of (3.8) is of low rank and the spectra of A and $-B$ are separated by a line. There are several other papers which present eigenvalue decay bounds for Sylvester equations of a certain structure, e.g., [4, 136, 157].

In Figure 3.2, the singular values of the solution of a Sylvester equation with A as in Example 2.3.18 and matrices F and G taken from the control problem introduced in Section 4.1.2 are depicted in decreasing order. It is seen that the singular values indeed decay rapidly; most of them are even smaller than the machine precision. Based on this observation, we modify the iteration scheme, as proposed in [23], see also [38, 15], as follows.

We introduce $n_\tau(M)$ as the numerical rank of a matrix M determined by a threshold τ, only writing n_τ whenever the meaning is clear. Exploiting the low-rank property of X, we compute factors $\tilde{Y}$ and $\tilde{Z}$ of a low-rank approximation to the solution X with $\tilde{Y} \in \mathbb{R}^{n \times n_\tau}$, $\tilde{Z} \in \mathbb{R}^{n_\tau \times m}$. In the following we will refer to these solution factors by the short denomination *approximate low-rank factors*.

We rewrite the Newton iteration (3.14) with $A_0 \leftarrow A$, $B_0 \leftarrow B$, $F_0 \leftarrow F$, $G_0 \leftarrow G$:

$$A_{j+1} \leftarrow \frac{1}{2}(A_j + A_j^{-1}), \tag{3.17a}$$

$$B_{j+1} \leftarrow \frac{1}{2}(B_j + B_j^{-1}), \tag{3.17b}$$

$$F_{j+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} F_j, & A_j^{-1} F_j \end{bmatrix}, \tag{3.17c}$$

$$G_{j+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} G_j \\ G_j B_j^{-1} \end{bmatrix}, \quad j = 0, 1, 2, \ldots, \tag{3.17d}$$

and get $Y = \frac{1}{\sqrt{2}} \lim_{j\to\infty} F_j$ and $Z = \frac{1}{\sqrt{2}} \lim_{j\to\infty} G_j$ as the factors of the solution $X = YZ$ in (3.16). This scheme is less expensive during the first steps if we assume that $p \ll n, m$. Since $F_j \in \mathbb{R}^{n\times 2^j p}$ and $G_j \in \mathbb{R}^{2^j p\times m}$, we have reduced storage requirements of $2^j\, p\, n + 2^j\, p\, m$ for $j < \log_2 \frac{nm}{p(n+m)}$ in the jth iteration step compared to a memory amount of nm for storing W_j in (3.14). The number of flops for the iteration parts (3.17c) and (3.17d) is changed from $2nm(n+m)$ in the W_j-iteration to $2 \cdot 2^j\, p(n^2 + m^2)$. Therefore the iteration scheme (3.17) is less expensive as long as j is sufficiently small. In the course of the iteration, this advantage gets lost as the number of columns of the F-iterates and the number of rows of the G-iterates is doubled in each step. As mentioned before, we expect that the solution has a low numerical rank; it can therefore be expected that the iterates are also of low numerical rank if the scheme converges quickly. To exploit this property and to avoid the exponential growth of the columns and rows, we apply a rank-revealing QR factorization (RRQR) [54] to F_{j+1}^T and G_{j+1} in each iteration step as proposed in [23]. We shortly review the definition and main properties of the RRQR factorization. For a given matrix $M \in \mathbb{R}^{n\times m}$ we assume without loss of generality that $n \geq m$. We define the numerical rank n_τ of M by a given threshold τ and by use of the ordered singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m \geq 0$ as follows:

$$\sigma_{n_\tau+1} < \sigma_1 \cdot \tau \leq \sigma_{n_\tau}.$$

Definition 3.1.5 *Let $M \in \mathbb{R}^{n\times m}$ have a QR factorization of the form*

$$M = QR\,\Pi = Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \Pi, \tag{3.18}$$

where Π is a permutation matrix, $Q \in \mathbb{R}^{n\times n}$ is an orthogonal matrix and $R \in \mathbb{R}^{n\times m}$ is upper triangular. An RRQR factorization *of M is defined by (3.18) if the numerical rank n_τ of M gives the order of $R_{11} \in \mathbb{R}^{n_\tau\times n_\tau}$ and if for the submatrices R_{11}, R_{22} the following properties are satisfied:*

$$\mathrm{cond}_2\,(R_{11}) := \|R_{11}\|_2\, \|R_{11}^{-1}\|_2 \lesssim \sigma_1/\sigma_{n_\tau}$$

and

$$\|R_{22}\|_2 = \sigma_{max}(R_{22}) \approx \sigma_{n_\tau+1}.$$

Then the condition number of R_{11} is bounded by $1/\tau$,

$$\mathrm{cond}_2\,(R_{11}) \leq 1/\tau,$$

and we can compress the rows of M by only considering entries in the upper part of the matrix R, that is the well conditioned part of M, neglecting the submatrix R_{22} of small norm. For the F-iterates the number of columns is compressed using a rank-revealing LQ decomposition (RRLQ) factorization (see, e.g., [74, Chapter 5]). The factorization of a matrix into a lower triangular matrix and an orthogonal matrix is equivalent to an RRQR factorization applied to its transpose. In our iteration scheme an RRQR factorization of G_{j+1} and an RRLQ factorization of F_{j+1} can be incorporated as follows:

$$\begin{aligned} F_{j+1}G_{j+1} &= \frac{1}{2}\left[\, F_j,\, A_j^{-1}F_j \,\right]\begin{bmatrix} G_j \\ G_jB_j^{-1} \end{bmatrix} \\ &= \frac{1}{2}\left[\, F_j,\, A_j^{-1}F_j \,\right] U R\, \Pi_G \\ &= \frac{1}{2}\underbrace{\left[\, F_j,\, A_j^{-1}F_j \,\right] U}_{\Pi_F L V}\begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}\Pi_G \\ &= \frac{1}{2}\,\Pi_F \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} V \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}\Pi_G, \end{aligned} \tag{3.19}$$

where Π_F and Π_G are permutation matrices, U and V are orthogonal. For simplicity, the numerical rank of G_{j+1} is now denoted by $r := n_\tau(G_{j+1})$ and $t := n_\tau(F_{j+1})$ is the numerical rank of F_{j+1}, both with respect to the same threshold τ. From the definition of the RRQR factorization it follows that $R_{11} \in \mathbb{R}^{r\times r}$ and $L_{11} \in \mathbb{R}^{t\times t}$. For $t \geq r$ we obtain approximate iterates $\tilde{F}_{j+1} \in \mathbb{R}^{n\times r}$ and $\tilde{G}_{j+1} \in \mathbb{R}^{r\times m}$ by truncating the matrices L and R as

$$\begin{aligned} \tilde{G}_{j+1} &:= \tfrac{1}{\sqrt{2}}\left[\, R_{11},\, R_{12} \,\right]\Pi_G, \\ \tilde{F}_{j+1} &:= \tfrac{1}{\sqrt{2}}\,\Pi_F \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix} V_{11}, \end{aligned} \tag{3.20}$$

where $V_{11} \in \mathbb{R}^{t\times r}$ denotes the upper left block of the orthogonal matrix V to adapt the matrix dimensions of the two factors.

Remark 3.1.6 For $t < r$ the factor $\tilde{F}_{j+1}$ is rank deficient using the above mentioned truncations. This can be corrected by multiplying $\tilde{G}_{j+1}$ from the left by $V_{11} \in \mathbb{R}^{t\times r}$ changing the truncation (3.20) to

$$\begin{aligned} \tilde{G}_{j+1} &:= \frac{1}{\sqrt{2}} V_{11}\left[\, R_{11},\, R_{12} \,\right]\Pi_G, \\ \tilde{F}_{j+1} &:= \frac{1}{\sqrt{2}}\,\Pi_F \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}. \end{aligned}$$

That is, the inner dimension of the product $\tilde{F}_{j+1}\tilde{G}_{j+1}$ is reduced to the numerical rank t of $\tilde{F}_{j+1}$. □

The iterates $\tilde{F}_{j+1}$ and $\tilde{G}_{j+1}$ have a reduced number of columns and rows, respectively, $\min\{r,t\}$ instead of $2^{j+1}\,p$, and we obtain approximate solution factors $\tilde{Y}$ and $\tilde{Z}$ by

$$\tilde{Y} = \frac{1}{\sqrt{2}} \lim_{j\to\infty} \tilde{F}_j, \quad \tilde{Z} = \frac{1}{\sqrt{2}} \lim_{j\to\infty} \tilde{G}_j.$$

Remark 3.1.7 The "optimal" norm scaling is adapted for the use in (3.17) by choosing

$$c_j^{(\text{opt.})} = \left(\frac{\sqrt{\|A_j^{-1}\|_2 \|B_j^{-1}\|_2}}{\sqrt{\|A_j\|_2 \|B_j\|_2}} \right)^{\frac{1}{2}},$$

thereby ignoring the contributions of the unavailable parts $F_j G_j$ in (3.12). Numerical experiments in [38] demonstrate the superiority of this choice compared to computations based on determinantal scaling. The iterates are scaled as follows:

$$A_j \to c_j A_j,\ B_j \to c_j B_j,\ F_j \to \sqrt{c_j} F_j,\ G_j \to \sqrt{c_j} G_j.$$

□

Complexity. The modified approach has several advantages. The storage requirements for X are reduced from nm in (3.14) to at most $nn_\tau(X)+mn_\tau(X)$ since the numerical ranks of the solution factors are bounded by $n_\tau(X)$. The expected low numerical rank of X reduces the overall workspace of the iteration to about

$$\mathcal{S}_{Mod.Sign}(n,m,X) = 2\left(n^2+m^2+2(n+m)n_\tau(X)\right)$$

real numbers. Furthermore, we have reduced computational costs of

$$\mathcal{N}_{Mod.Sign}(n,m,X) = 2n^3+2m^3+2(n^2+m^2)n_\tau(X)$$

flops for one iteration step in (3.17).

Remark 3.1.8 So far, we have neglected the complexity of the RRQR factorization. We use a block QR factorization with pivot windows followed by a variant of an algorithm suggested by Chandrasekaran and Ipsen [55]. The block algorithm is proposed in [41] and it is shown that the performance is faster than for the standard LAPACK routine DGEQPF for computing QR factorizations with column pivoting. The complexity of the computational steps involved to compute low-rank factors $\tilde{G}_j$, $\tilde{F}_j$ by standard RRQR implementations is described in the following. We denote the number of rows of G_j and of columns of F_j by p_j. The aim is to use the RRQR factorizations to keep this number small. The factorization of $G_j \in \mathbb{R}^{p_j\times m}$ with $r = \operatorname{rank}(G_j)$ by the pivoted Householder QR algorithm requires $4\,p_j mr - 2r^2(p_j+m) + 4\frac{r^3}{3}$ flops [74, Algorithm 5.4.1]. The explicit computation of the orthogonal factor U is avoided, it is applied "on the fly" to F_j which requires $2nr(2\,p_j - r)$ flops [74, Section 5.1.6]. For $F_j \in \mathbb{R}^{n\times p_j}$, assuming $t = \operatorname{rank}(F_j)$, the pivoted LQ factorization of F_jU is computed by additional $4\,p_jnt - 2t^2(p_j+n) + 4\frac{t^3}{3}$ flops. Since

the next iterate $\tilde{F}$ (for $t \leq r$) is obtained by a multiplication with the submatrix $V_{11} \in \mathbb{R}^{t \times r}$, the orthogonal matrix V is computed explicitly. This takes $2t^2 p_j - 2\frac{t^3}{3}$ flops, see [74, Algorithm 5.1.2]. The last step is the computation of the matrix-matrix product which requires further $2ntr$ flops. □

3.2 Lyapunov Equations

Reducing the dimension of a large-scale LTI system by balanced truncation requires the solution of two large-scale Lyapunov equations. First, we consider Lyapunov equations of a general form, given by

$$AX + XA^T + W = 0, \tag{3.21}$$

with $A \in \mathbb{R}^{n \times n}$, symmetric $W \in \mathbb{R}^{n \times n}$ and unknown $X \in \mathbb{R}^{n \times n}$. This is the symmetric variant of the Sylvester equation (3.8) with $B = A^T$. We will assume that the spectrum of A is contained in the open left half complex plane $\mathbb{C}^-$. This type of Lyapunov equations is also called *stable Lyapunov equation* [92, 95] and admits a unique symmetric solution X. A standard direct method for solving Lyapunov equations is the Bartels-Stewart method [14] as already mentioned in Section 3.1 for the solution of Sylvester equations. The method can be modified to take advantage of the symmetry in (3.21) resulting in a reduced complexity of

$$\mathcal{N}_{BS}(n) = 32n^3$$

and reduced storage requirements of size $3n^2$ under the assumption that A is overwritten with its Schur form and W with the solution.

3.2.1 The Sign Function Iteration

Numerical methods for Lyapunov equations can be based on the sign function method introduced in Section 3.1 for the solution of Sylvester equations. In order to solve the Lyapunov equation (3.21), we apply this iteration to

$$Z = \begin{bmatrix} A & W \\ 0 & -A^T \end{bmatrix}, \tag{3.22}$$

and obtain the following iteration scheme

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{j+1} &\leftarrow \frac{1}{2}(Z_j + Z_j^{-1}) \\ &= \begin{bmatrix} \frac{1}{2}(A_j + A_j^{-1}) & \frac{1}{2}(W_j + A_j^{-1} W_j A_j^{-T}) \\ 0 & -\frac{1}{2}(A_j + A_j^{-1})^T \end{bmatrix}, \quad j = 0, 1, 2, \ldots. \end{aligned}$$

The solution X of (3.21) is given by dividing the upper right block in

$$\operatorname{sign}(Z) = \lim_{j \to \infty} Z_j = \begin{bmatrix} -I_n & 2X \\ 0 & I_n \end{bmatrix}$$

by 2. To accelerate the initial convergence, some of the iterates can be scaled,

$$Z_{j+1} \leftarrow \frac{1}{2}(c_j Z_j + \frac{1}{c_j} Z_j^{-1}),$$

where $c_j > 0$ are suitably chosen parameters as described in Section 3.1.1 for the more general Sylvester equations.

The stopping criterion (3.15) for the Sylvester equation from Section 3.1.1 simplifies to

$$\|A_j + I_n\| \leq \text{tol}. \tag{3.23}$$

With two additional iteration steps and an appropriate choice of norm and relaxed tolerance, we usual get a high accuracy in the size of machine precision due to the quadratic convergence, see [30] for details.

Complexity. Storage requirements and number of flops are equal to the corresponding parts in (3.14). Note that due to the special form of the Lyapunov equation the B_j-iteration is redundant. We get an overall complexity of

$$\mathcal{N}_{Sign}(n) = 6n^3$$

for the sign function iteration to solve the symmetric problem (3.21).

3.2.2 Factorized Solution of Lyapunov Equations

We restrict our attention to the Lyapunov equation for computing the controllability Gramian X of a LTI system as introduced in Chapter 2,

$$AX + XA^T + BB^T = 0, \tag{3.24}$$

with stable $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$, $m \ll n$. The stability of A together with the positive semi-definiteness of the "right-hand side term" BB^T imply that the Lyapunov equation (3.24) has a unique, symmetric nonnegative definite solution X [113]. Hence, it can be factorized as $X = YY^T$. Possibilities for Y are the Cholesky factor of X, i.e., $Y \in \mathbb{R}^{n\times n}$ is a square lower triangular matrix, and a full-rank factor of X, i.e., $Y \in \mathbb{R}^{n\times \text{rank}(X)}$ is a rectangular matrix. The latter option is of particular interest for large-scale computations if X has low rank, $\text{rank}\,(X) \ll n$, as (3.24) represents a linear system of equations with $n(n+1)/2$ unknowns (exploiting symmetry). In such a situation, the memory requirements for storing X can be considerably reduced by working with Y instead of X. Interpreting the rank of X as numerical rank, it is often the case that this numerical rank is very low even though theoretically, X may be nonsingular. In that case, using a spectral (or singular value) decomposition of X, it is easy to see that Y can be approximated by a "tall" matrix $\tilde{Y} \in \mathbb{R}^{n\times n_\tau}$, so that

$$\frac{\|X - \tilde{Y}\tilde{Y}^T\|_2}{\|X\|_2} \leq \tau$$

with the tolerance threshold τ determining the numerical rank n_τ of X with $n_\tau \ll n$. As for Sylvester equations, it can often be observed that in many

large-scale applications the eigenvalues of X decay rapidly, so that a low-rank approximation in the form described above exists; see [4, 76, 136].

To obtain the factorized solution, we initialize the iteration by $A_0 \leftarrow A$, $B_0 \leftarrow B$ and partition (3.23) into two parts,

$$A_{j+1} \leftarrow \frac{1}{2}(A_j + A_j^{-1}), \tag{3.25a}$$

$$B_{j+1} \leftarrow \frac{1}{\sqrt{2}} \left[B_j,\ A_j^{-1} B_j \right], \quad j = 0, 1, 2, \ldots. \tag{3.25b}$$

We directly iterate on the factors and obtain by $Y = \frac{1}{\sqrt{2}} \lim_{j\to\infty} B_j$ a factor of the solution

$$X = YY^T = \frac{1}{2} \lim_{j\to\infty} B_j B_j^T.$$

As in Section 3.1.2, it can be observed that the size of the matrix B_{j+1} in (3.25b) is doubled in each iteration step. Therefore, it is proposed in [30] to apply an RRLQ to B_{j+1} in order to limit the exponentially growing number of columns:

$$B_{j+1} = \Pi L\, Q = \Pi \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} Q.$$

Here Π is a permutation matrix, Q is orthogonal, and the numerical rank is denoted by $r := n_\tau(B_{j+1})$. Then, L_{11} is a $r \times r$ matrix, while L_{22} is of small norm,

$$\|L_{22}\|_2 < \tau \|B_{j+1}\|_2,$$

so that it can be neglected. Only the entries in the left r columns of L have to be stored,

$$\tilde{B}_{j+1} := \Pi \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}, \tag{3.26}$$

for obtaining an approximate solution $\tilde{Y} = \frac{1}{\sqrt{2}} \lim_{j\to\infty} \tilde{B}_j$. The symmetric structure of the Lyapunov equation provides first insights in how to choose the parameter τ for the sign function iteration. For simplicity we assume that the LQ factorization can be computed without pivoting, i.e. $\Pi = I_n$. By a simple calculation

$$\begin{aligned} X - \frac{1}{2} B_{j+1} B_{j+1}^T &= X - \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} QQ^T \begin{bmatrix} L_{11}^T & L_{21}^T \\ 0 & L_{22}^T \end{bmatrix} \\ &= X - \begin{bmatrix} L_{11}L_{11}^T & L_{11}L_{21}^T \\ L_{21}L_{11}^T & L_{21}L_{21}^T + L_{22}L_{22}^T \end{bmatrix} \\ &= X - \frac{1}{2} \tilde{B}_{j+1} \tilde{B}_{j+1}^T - \begin{bmatrix} 0 & 0 \\ 0 & L_{22}L_{22}^T \end{bmatrix} \end{aligned}$$

and under the assumption

$$\frac{1}{2} \|B_{j+1}\|_2^2 \approx \|X\|_2,$$

which is justified since $\frac{1}{2}\lim_{j\to\infty} B_j B_j^T = X$, we obtain the following estimate for the approximation errors of $B_{j+1}B_{j+1}^T$ and $\tilde{B}_{j+1}\tilde{B}_{j+1}^T$:

$$\frac{\|X - \frac{1}{2}\tilde{B}_{j+1}\tilde{B}_{j+1}^T\|_2}{\|X\|_2} \leq \frac{\|X - \frac{1}{2}B_{j+1}B_{j+1}^T\|_2}{\|X\|_2} + \tau^2.$$

This shows that if we choose the numerical rank threshold τ of order

$$\tau \sim \sqrt{\text{EPS}},$$

then the relative error introduced by row compression in (3.26) is of order EPS and therefore negligible. This motivates the choice of τ in the next chapter, where we introduce further approximation errors using the hierarchical matrix format.

Remark 3.2.1 We introduce scaling to attain the domain of quadratic convergence faster. In the partitioned iteration scheme (3.25) scaling is incorporated in the following way:

$$\begin{aligned} A_{j+1} &\leftarrow \frac{1}{2}(c_j A_j + \frac{1}{c_j}A_j^{-1}), \\ B_{j+1} &\leftarrow \frac{1}{\sqrt{2}}\left[\ \sqrt{c_j}B_j,\ \ \tfrac{1}{\sqrt{c_j}}A_j^{-1}B_j\ \right], \end{aligned}$$

where c_j is an appropriate scaling parameter. A choice of c_j is discussed in [96, 75]. We will use a problem adapted variant of the norm scaling as suggested in [38], see also Remark 3.1.7,

$$c_j^{(\text{opt.})} = \sqrt{\frac{\|A_j^{-1}\|_2}{\|A_j\|_2}},$$

which is supposed to be superior to other parameter choices as already observed for the solution of Sylvester equations. □

Complexity. Comparable to the computation of low-rank factors of a Sylvester equation as described in Section 3.1.2, computing a low-rank solution factor $\tilde{Y}$ of a Lyapunov equation reduces the storage requirements for the solution from $n(n+1)/2$ for X to $nn_\tau(X)$ for $\tilde{Y}$. The complete storage demand for the modified sign function iteration is about $2n^2 + 4nn_\tau(X)$ real numbers.

We observe a reduced computational complexity for the B_j-iteration of about $2n^2r$ flops versus $4n^3$ flops for the W_j-iteration in Section 3.2.1. We sum up all flops and obtain

$$\mathcal{N}_{Mod.Sign}(n, X) = 2n^3 + 2n^2 n_\tau(X)$$

as the overall complexity of the iteration scheme (3.25).

Remark 3.2.2 For $B_j \in \mathbb{R}^{n\times m_j}$ with $r = \text{rank}(B_j)$ and m_j denoting the (compressed) number of columns of the jth iterate, the complexity of computing an

RRQR is $4\,m_j nr - 2r^2(m_j + n) + 4\frac{r^3}{3}$ flops. In contrast to the rank-revealing steps (3.19) in the nonsymmetric sign function iteration (3.17) for Sylvester equations, in the RRQR factorization $B_{j+1} = \Pi L\,Q$ for the computation of $B_{j+1}B_{j+1}^T$, the orthogonal matrix Q does not have to be stored, see (3.26). Therefore, the complexity of computing column compressions for the solution of Lyapunov equations is reduced compared to the complexity of the compression steps for the solution of Sylvester equations as described in Remark 3.1.8. □

3.2.3 Factorized Solution of Generalized Lyapunov Equations

In this section, we consider generalized Lyapunov equations of the form

$$AXE^T + EXA^T + BB^T = 0, \tag{3.27}$$

where $A, E \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. Such matrix equations are associated with linear, time-invariant systems in generalized state space form (2.2). Note that (3.27) reduces to a standard Lyapunov equation if $E = I_n$. Generalized Lyapunov equations with $E \neq I_n$ play an important role in various fields related to descriptor systems [57], such as minimal realization or balanced truncation model reduction [159]. In the following, we assume that E is nonsingular and that $A - \lambda E$ is a stable matrix pencil. Under these assumptions, the generalized Lyapunov equation (3.27) has a unique symmetric, positive semi-definite solution X. Gardiner and Laub [69] proposed an extension of the sign function iteration for the solution of generalized algebraic Riccati equations. From this approach a generalization of the iteration scheme for the numerical solution of (3.27) can simply be derived. Instead of the single matrix Z in (3.22), the matrix pencil

$$Z - \lambda Y = \begin{bmatrix} A & BB^T \\ 0 & -A^T \end{bmatrix} - \lambda \begin{bmatrix} E & 0 \\ 0 & E^T \end{bmatrix} \tag{3.28}$$

is considered. Theoretically, the solution of (3.27) can be obtained from the $(1,2)$ block of $\operatorname{sign}(Y^{-1}Z)$. Applying the standard sign function iteration directly to $Y^{-1}Z$, however, has the disadvantage that the possibly ill-conditioned matrix E has to be inverted for starting the iteration. The approach in [69] avoids this drawback using the so-called *generalized Newton iteration*

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{j+1} &\leftarrow \frac{1}{2}(Z_j + YZ_j^{-1}Y), \quad j = 0, 1, 2, \ldots. \end{aligned}$$

It can be easily seen that if $\tilde{Z}_j$ denotes the jth iterate of the standard sign function iteration applied to $Y^{-1}Z$, then $Z_j = Y\tilde{Z}_j$. This implies that the iteration converges under the given assumptions to

$$\lim_{j\to\infty} Z_j = Y \cdot \operatorname{sign}(Y^{-1}Z).$$

In [30] it is shown that this scheme significantly simplifies when applied to a matrix pencil of the form (3.28):

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{j+1} &\leftarrow \frac{1}{2}(Z_j + YZ_j^{-1}Y) \\ &= \begin{bmatrix} \frac{1}{2}(A_j + EA_j^{-1}E) & \frac{1}{2}(B_jB_j^T + EA_j^{-1}B_jB_j^TA_j^{-T}E^T) \\ 0 & -\frac{1}{2}(A_j^T + E^TA_j^{-T}E^T) \end{bmatrix}. \end{aligned}$$

The solution X of (3.27) can then be obtained by noting that

$$\lim_{j\to\infty} Z_j = \begin{bmatrix} -E & 2EXE^T \\ 0 & E^T \end{bmatrix}. \tag{3.29}$$

Just as in the previous sections for Sylvester and standard Lyapunov equations, we are interested in computing a factor Y of the solution X, $X = YY^T$, instead of the solution itself. Therefore, we consider the iteration in factorized form as introduced in [30] for $E \neq I_n$ with $A_0 \leftarrow A$, $B_0 \leftarrow B$:

$$A_{j+1} \leftarrow \frac{1}{2}(A_j + EA_j^{-1}E), \tag{3.30a}$$

$$B_{j+1} \leftarrow \frac{1}{\sqrt{2}}\begin{bmatrix} B_j, & EA_j^{-1}B_j \end{bmatrix}, \quad j = 0, 1, 2, \ldots. \tag{3.30b}$$

We obtain $Y = \frac{1}{\sqrt{2}}E^{-1}\lim_{j\to\infty} B_j$ as a solution factor of (3.27):

$$X = YY^T = \frac{1}{2}E^{-1}\lim_{j\to\infty}(B_jB_j^T)E^{-T}.$$

It can be seen from (3.29) that $\lim_{j\to\infty} A_j = -E$, so we suggest as natural stopping criterion:

$$\|A_j + E\| \leq \mathrm{tol} \cdot \|E\|. \tag{3.31}$$

As proposed in [30], in general we reach the desired accuracy by a suitable choice of tol, performing two further iteration steps once the criterion is satisfied. For computing approximate low-rank factors of the solution X we compress the columns of B_{j+1} using an RRLQ factorization as described in Section 3.2.2 for the standard case.

Remark 3.2.3 In order to accelerate convergence, the iterates A_j and B_j will be replaced by c_jA_j, $\sqrt{c_j}B_j$ leading to the following scheme with scaling:

$$\begin{aligned} A_{j+1} &\leftarrow \frac{1}{2}(c_jA_j + \frac{1}{c_j}EA_j^{-1}E), \\ B_{j+1} &\leftarrow \frac{1}{\sqrt{2}}\begin{bmatrix} \sqrt{c_j}B_j, & \frac{1}{\sqrt{c_j}}EA_j^{-1}B_j \end{bmatrix}. \end{aligned}$$

A proposed choice of scaling parameter in [69] is a generalization of the determinantal scaling

$$c_j^{(\mathrm{det.})} = \frac{|\det(E)|^{1/n}}{|\det(A_j)|^{1/n}}.$$

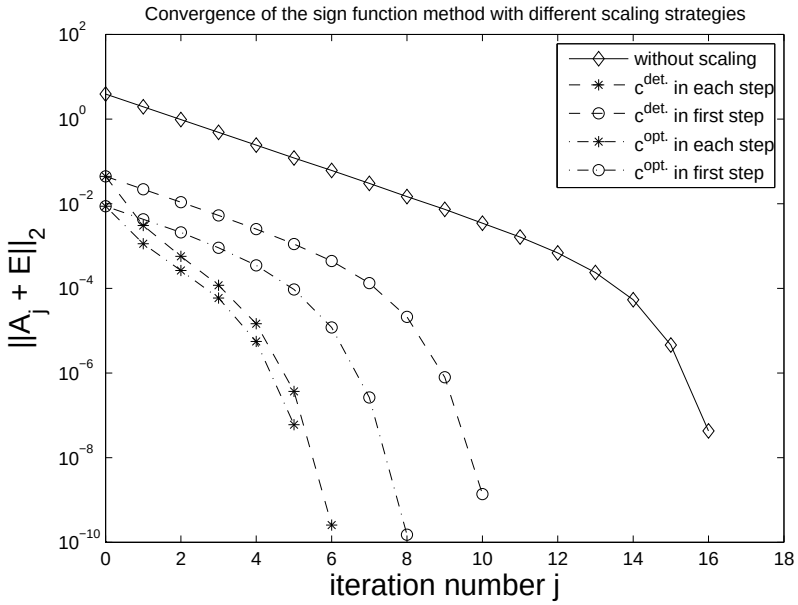


Figure 3.3: Convergence history for sign(A) with $n = 1024$.

In our numerical experiments in Section 4.3.2 we will use the following choice:

$$c_j^{(\text{opt.})} = \sqrt{\frac{\|EA_j^{-1}E\|_2}{\|A_j\|_2}}.$$

This is inspired by the motivation for the scaling in the standard case, the numerical results confirm its ability to accelerate the convergence. For a typical convergence history of the scaled sign function iteration see Figure 3.3. Here E is the mass, A the stiffness matrix from the FEM discretization of the two-dimensional heat equation as in Example 2.3.18 with $n = 1024$. As tolerance for the stopping criterion (3.31) we set

$$\text{tol} = 10\, n\, \sqrt{\text{EPS}}.$$

It is observed that the best performance is achieved by applying the "optimal" norm scaling in each iteration step. □

Complexity. The complexity for one iteration step of the modified iteration scheme (3.30) is given by

$$\mathcal{N}_{Mod.Sign}(n, X) = 6n^3 + 4n^2 n_\tau(X)$$

flops and we need about

$$\mathcal{S}_{Mod.Sign}(n, X) = 3n^2 + 4nn_\tau(X)$$

real number storage for the iteration scheme neglecting the costs of computing RRLQ factorizations.

3.3 Stein Equations

We will now consider the discrete-time variant of the Lyapunov equation, the so called *Stein equation*, skipping the more general form, the discrete Sylvester equation. These linear matrix equations are fundamental in linear control theory of discrete-time, linear, time-invariant systems. For instance, reducing the order of a large-scale discrete-time system by balanced truncation requires the solution of two Stein equations similar to the solution of two Lyapunov equations in the continuous-time case. We consider Stein equations of the form

$$AXA^T - X + W = 0 \tag{3.32}$$

with $A \in \mathbb{R}^{n\times n}$, $W \in \mathbb{R}^{n\times n}$, $W = W^T$ and the unknown matrix $X \in \mathbb{R}^{n\times n}$. Similar to (3.9) we can reformulate the equation such that (3.32) has a unique symmetric solution, if and only if,

$$\lambda_i\lambda_j \neq 1, \quad i, = 1, \ldots, n,\ j = 1, \ldots, n.$$

Then, the system matrix of the linear equation

$$(A \otimes A - I_n)\,\vec{X} = -\vec{W},$$

is nonsingular, and existence and uniqueness of X guaranteed. For $\rho(A) < 1$ this condition is fulfilled and we call (3.32) stable. Schur stability of A will be assumed in the following. For stable Stein equations, X is positive semi-definite as can be shown from the Lyapunov stability theory, e.g., in [113], or from the explicit solution formula

$$X = \sum_{j=0}^{\infty} A^j W (A^T)^j, \tag{3.33}$$

as this power series converges when $\rho(A) < 1$. For the numerical solution of Stein equations we have to note that even if we exploit the symmetry of X we have to solve a system with $n(n+1)/2$ unknowns. There are several direct methods for the numerical solution of Stein equations based on transforming the matrix A to quasi-upper triangular form using the QR algorithm, see [13, 73, 149]. These methods are restricted to problems of moderate size because of their computational complexity and memory requirements. In [35], iterative solvers are proposed which are easy to parallelize and are well suited for problems of larger sizes. We will review some of the results with focus on the squared Smith iteration which will be adapted for the purpose of model reduction.

3.3.1 The Squared Smith Iteration

As a simple example for an iterative scheme suitable for the numerical solution of Stein equations we consider the fixed point iteration initialized by $X_0 \leftarrow W$:

$$X_{j+1} \leftarrow AX_jA^T + W, \quad j = 0, 1, 2 \ldots.$$

The iteration converges linearly to X if $\rho(A) < 1$. A quadratically convergent version of this fixed point iteration scheme recommended in [152] is the *squared Smith iteration*. Setting $A_0 \leftarrow A$, $X_0 \leftarrow W$, the iteration can be written as

$$X_{j+1} \leftarrow A_j X_j A_j^T + X_j, \tag{3.34a}$$
$$A_{j+1} \leftarrow A_j^2, \qquad j = 0, 1, 2 \ldots. \tag{3.34b}$$

Remark 3.3.1 The squared Smith iteration converges globally quadratically for Stein equations with Schur stable matrix A. It is derived in [152] that for $\rho(A) < 1$ there exist real constants $M > 0$ and $0 < r < 1$ with

$$\|X - X_j\|_2 \leq M \|W\|_2 (1 - r)^{-1} r^{2^j}.$$

Nevertheless, for safely avoiding overflow caused by increasing $\|A_j\|_2$, one should ensure that $\|A\|_F < 1$, see [35] for details. □

It follows that the solution of (3.32) is given by

$$X = \lim_{j \to \infty} X_j$$

whereas for the other part of the iteration we have $\lim_{j\to\infty} A_j = 0$, caused by the Schur stability of the matrix A. This convergence provides a simple stopping criterion for the squared Smith iteration

$$\|A_j\| \leq \text{tol}, \tag{3.35}$$

with a suitably chosen bound tol ≥ 0. The computational complexity of this iteration scheme comes from the multiplication of $n \times n$ matrices. Exploiting the symmetry of the X_j-iterates, the complexity is reduced to

$$\mathcal{N}_{\text{Smith}}(n) = 3n^3$$

rather than $4n^3$ flops for general matrix products.

Remark 3.3.2 It is also possible to transform the Stein equation (3.32) into a Lyapunov equation using a Cayley transformation $c(\cdot)$ applied to the matrix A in (3.32),

$$c(A) = (A - I_n)^{-1}(A + I_n).$$

The resulting Lyapunov equation

$$\hat{A}X + X\hat{A}^T + \hat{W} = 0 \tag{3.36}$$

is given by the matrices

$$\begin{aligned} \hat{A} &= c(A), \\ \hat{W} &= 2(A - I_n)^{-1} W (A - I_n)^{-T}. \end{aligned}$$

Therefore, we can also solve the Stein equation (3.32) using the sign function iteration for the solution of (3.36) as described in Section 3.2.1. □

3.3.2 Factorized Solution of Stein Equations

We consider a Stein equation with the constant term given in factorized form. These equations appear in a wide range of applications, for instance the controllability Gramian of a linear discrete-time system is given as the unique, symmetric solution of the following Stein equation

$$AXA^T - X + BB^T = 0, \tag{3.37}$$

with Schur stable $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and unknown matrix $X \in \mathbb{R}^{n\times n}$. As for the previous types of matrix equations, we are interested in computing Cholesky or full-rank factors of the solution, i.e., $X = YY^T$. Note that for stable systems the solution X is positive semi-definite and hence can always be factorized in this form. If $\operatorname{rank}(X) \ll n$ the storage requirements can be reduced by computing a full-rank factor $Y \in \mathbb{R}^{n\times \operatorname{rank}(X)}$. From the explicit solution formula (3.33) at least a small numerical rank of X can be expected due to the fast convergence of the power iteration for Schur stable A [74, Section 7.3]. To obtain low-rank approximations to the full-rank solution factors, we consider the following problem adapted variant as proposed in [35] with $B_0 \leftarrow B$, $A_0 \leftarrow A$,

$$B_{j+1} \leftarrow \begin{bmatrix} B_j, & A_jB_j \end{bmatrix}, \tag{3.38a}$$

$$A_{j+1} \leftarrow A_j^2, \qquad j = 0, 1, 2 \ldots. \tag{3.38b}$$

The matrix $Y = \lim_{j\to\infty} B_j$ is a factor of the solution

$$X = YY^T = \lim_{j\to\infty} B_jB_j^T.$$

Since the size of the matrix B_{j+1} in (3.38a) is doubled in each iteration step we apply an RRLQ factorization to B_{j+1}. We have seen this computational step already for the computation of approximate low-rank factors of Sylvester and Lyapunov equations. In each iteration step we compute

$$B_{j+1} = \Pi LQ = \Pi \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} Q$$

with a permutation matrix Π and an orthogonal matrix Q. The matrix L is lower triangular. By a given threshold τ we determine the numerical rank of the B_j-iterates and denote them by $r := n_\tau(B_{j+1})$. The matrix block L_{11} is of size $r \times r$ and L_{22} is of small norm. We compress the number of columns by storing only the entries in the left part of L. The solution factor can be written as $\tilde{Y} = \lim_{j\to\infty} \tilde{B}_j$ by use of the new iterate

$$\tilde{B}_{j+1} := \Pi \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}.$$

Remark 3.3.3 (**Stopping criterion**) Using an RRLQ factorization in each iteration step provides another way (besides (3.35)) to stop the iteration. Since

the singular values of the B-iterates are computed for the numerical rank decision we suggest to stop the iteration if the change in the largest singular values σ_1 stagnates. That is, if the relative error is bounded

$$\frac{|\sigma_1(B_j) - \sigma_1(B_{j+1})|}{\sigma_1(B_j)} < \text{tol}.$$

This heuristic criterion ensures a convergence of at least the largest singular value of the solution factor. For a suitable choice of the tolerance we refer to the numerical experiments in Section 4.4.2. □

Complexity. If we count the flops needed in each iteration step of (3.38) we observe a reduced amount of

$$\mathcal{N}_{\text{Mod.Smith}}(n, X) = n^3 + n^2 n_\tau(X)$$

flops for computing B_{j+1} compared to $3n^3$ flops in (3.34) for computing X_{j+1}. We exclude the costs of the RRQR factorization. The storage requirements for the solution are reduced from $n(n+1)/2$ for symmetric X to $n_\tau(X) \times n$ for storing $\tilde{S}$ as low-rank factor.

Chapter 4

Solvers Based on Data-Sparse Approximation

In the previous chapter we have seen several iteration schemes for the numerical solution of linear matrix equations. All these methods have been modified for computing the solution in factorized form resulting in reduced storage requirements and lower computational complexity. But as can be seen from the complexity estimates of each modified iteration throughout Chapter 3, the amount of storage still grows quadratically with the order of possibly large coefficient matrices. Furthermore, the inversion and multiplication of the large-scale iterates involves a cubically growing demand of flops.

In [79], the sign function method for solving the more general algebraic Riccati equation (ARE) is combined with a data-sparse matrix representation and a corresponding approximate arithmetic. The method computes the solution of an ARE in $\mathcal{H}$-matrix format with linear-polylogarithmic complexity and is therefore suitable for large-scale problems. It can be adapted rather directly to the solution of Sylvester or Lyapunov equations.

We need solutions of matrix equations as a first step in model order reduction methods for large-scale LTI systems. From these solutions, either a SVD or a spectral decomposition is required for the computation of the projection matrices. Since these factorizations are not available in the $\mathcal{H}$-matrix format, we compute approximate (dense) low-rank factorizations of the solutions of the matrix equations. To make the methods applicable to large-scale problems, we propose new $\mathcal{H}$-matrix arithmetic based iteration schemes in partitioned form. We integrate the data-sparse hierarchical matrix format in the expensive parts of all modified methods. With the corresponding formatted arithmetic we obtain solvers of linear-polylogarithmic complexity for a large class of practically relevant matrices which are particularly efficient in model reduction. Previous work on solvers based on the hierarchical matrix arithmetic is published in [15] for the factorized solution of Sylvester equations and in [16, 17] for the factorized solution of Lyapunov equations. Some of the results are reviewed in the corresponding sections. In this chapter we rewrite all modified iteration schemes from Chapter 3 with data-sparse approximations. We examine the performance of the new solvers both theoretically and by presenting meaningful numerical

experiments, thereby demonstrating the usefulness of the solvers for large-scale computation.

4.1 $\mathcal{H}$-Matrix Arithmetic Based Sign Function Iteration for Sylvester Equations

We consider the sign function iteration as presented in Section 3.1.2 for the solution of the Sylvester equation

$$AX + XB + FG = 0.$$

To make the method feasible for large-scale computations we employ the data-sparse $\mathcal{H}$-matrix format and the corresponding formatted arithmetic. Grasedyck shows in [76] that in many practical applications the solution of a Sylvester equation can be approximated in $\mathcal{H}$-matrix format. Furthermore, in [78] a multigrid algorithm is used for solving Sylvester equations with solutions in data-sparse format and [79] computes $\mathcal{H}$-matrix solutions of an ARE based on the sign function iteration with formatted arithmetic. Our approach is different: we compute the solutions in dense low-rank format since this format is suitable for model order reduction. Instead of $X \in \mathbb{R}^{n\times m}$, we compute approximate low-rank solution factors $\tilde{Y} \in \mathbb{R}^{n\times n_\tau(X)}$ and $\tilde{Z} \in \mathbb{R}^{n_\tau(X)\times m}$. Therefore, the storage requirements as well as the computational complexity are reduced in part (3.17c) and (3.17d) of the modified iteration scheme compared to the original sign function iteration (3.14). But the amount of storage is still quadratic and the complexity cubic in the size of the intermediate matrices A_j and B_j in the iteration parts (3.17a) and (3.17b). Moreover, even if the system matrices A and B are sparse, resulting, e.g., from finite element discretization of elliptic partial differential operators, a large amount of memory is required during the Newton iteration caused by fill-in during the matrix inversion. Note that also the factors L and U from the LU-decomposition of a sparse matrix are usually fully populated. To avoid this effect, the large-scale iterates A_j and B_j are approximated in the data-sparse $\mathcal{H}$-matrix format and the corresponding approximate arithmetic is used to reduce the computational cost in the corresponding parts of the iteration scheme. Details concerning the $\mathcal{H}$-matrix format and arithmetic can be found in Chapter 2.

We stress that a necessary condition for integrating $\mathcal{H}$-matrix techniques into the iteration scheme is the representability of the iterates as $\mathcal{H}$-matrices. In many applications as boundary element discretization of non-local integral operators and for finite element discretization of elliptic partial differential operators, the resulting matrices allow for a data-sparse representation [89, 91]. In [20, 22] it is proven that for elliptic partial differential operators with L^∞-coefficients and for inverses of FEM matrices, blockwise low-rank approximations exist. Furthermore, control problems based on elliptic operators can be treated by hierarchical matrix approximations [79].

We assume that the square system matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{m\times m}$ can be approximated in the set of $\mathcal{H}$-matrices $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, \epsilon)$ where ϵ denotes the

blockwise accuracy of the adaptive arithmetic (see end of Section 2.3). That is, the iteration can be started with

$$A_0 = A_{\mathcal{H}}, \quad B_0 = B_{\mathcal{H}},$$

where for some $c > 0$ the approximation errors are bounded [75, Chapter 7] as

$$\begin{aligned} \|A - A_{\mathcal{H}}\|_2 &\leq c \log_2(n)\, \epsilon \max_{r\times s \in \mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})} \|A_{|r\times s}\|_2, \\ \|B - B_{\mathcal{H}}\|_2 &\leq c \log_2(m)\, \epsilon \max_{r\times s \in \mathcal{L}^+(T_{\mathcal{I}\times\mathcal{I}})} \|B_{|r\times s}\|_2. \end{aligned}$$

We replace the usual addition and inversion in (3.17a) and (3.17b) by formatted arithmetic as introduced in Section 2.3. For the approximate matrix inversion, $\mathcal{H}$-LU factorizations of the matrices A_j and B_j are computed. With $\mathcal{H}$-based forward/backward substitution the approximate inverses $A_{\mathcal{H},j}^{-1}$ and $B_{\mathcal{H},j}^{-1}$ are obtained, see Algorithm 3, and the intermediates A_j and B_j are computed in the set of $\mathcal{H}$-matrices,

$$\begin{aligned} A_{j+1} &\leftarrow \frac{1}{2}(A_j \oplus A_{\mathcal{H},j}^{-1}), \\ B_{j+1} &\leftarrow \frac{1}{2}(B_j \oplus B_{\mathcal{H},j}^{-1}). \end{aligned}$$

Using the adaptive $\mathcal{H}$-matrix arithmetic with accuracy ϵ (instead of a given fixed rank k) in each matrix block, ensures that errors caused by the formatted inversion and addition are bounded by the adaptive rank choice. These blockwise ranks should not increase "too much" during the sign function iteration so that an efficient implementation is guaranteed.

The matrices F_j and G_j, which yield the solution factors at the end of the iteration, are stored in the usual dense "full" format. Incorporating the formatted inverses into the iteration scheme changes (3.17c) and (3.17d) to

$$\begin{aligned} F_{j+1} &\leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} F_j, & A_{\mathcal{H},j}^{-1} F_j \end{bmatrix}, \\ G_{j+1} &\leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} G_j \\ G_j B_{\mathcal{H},j}^{-1} \end{bmatrix}. \end{aligned}$$

In these iteration parts, the product of $\mathcal{H}$- and "full" matrices is computed by a formatted matrix-vector product. To limit the increasing number of columns and rows of the two solution factors, an RRQR (resp. RRLQ) factorization is used as described in Section 3.1.2.

We summarize in Algorithm 4 all computational steps for computing approximate low-rank factors for the solution of (3.16).

Algorithm 4 Calculate low-rank factors $\tilde{Y}$, $\tilde{Z}$ of X for $AX + XB + FG = 0$.

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $F \in \mathbb{R}^{n\times p}$, $G \in \mathbb{R}^{p\times m}$, tol, ϵ, τ

OUTPUT: Approximate low-rank factors Y and Z of the solution X.

$A_0 \leftarrow (A)_{\mathcal{H}}$, $B_0 \leftarrow (B)_{\mathcal{H}}$

$F_0 \leftarrow F$, $G_0 \leftarrow G$

$j = 0$

while $\max\{\|A_j + I_n\|_2, \|B_j + I_m\|_2\} > \text{tol}$ **do**

  Compute the approximate inverse $A^{-1}_{\mathcal{H},j}$ by Algorithm 3.

  $A_{j+1} \leftarrow \frac{1}{2}(A_j \oplus A^{-1}_{\mathcal{H},j})$

  $F_{j+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} F_j, & A^{-1}_{\mathcal{H},j} F_j \end{bmatrix}$

  Compute the approximate inverse $B^{-1}_{\mathcal{H},j}$ by Algorithm 3.

  $B_{j+1} \leftarrow \frac{1}{2}(B_j \oplus B^{-1}_{\mathcal{H},j})$

  $G_{j+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} G_j \\ G_j B^{-1}_{\mathcal{H},j} \end{bmatrix}$

  Compute an RRQR factorization

$$G_{j+1} = U \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \Pi_G$$

  with $\|R_{22}\|_2 < \tau \|G_{j+1}\|_2$ and $R_{11} \in \mathbb{R}^{r\times r}$.

  Compress rows of G_{j+1} to size r:

$$G_{j+1} \leftarrow [\, R_{11}, R_{12} \,] \Pi_G.$$

  Compute an RRLQ factorization

$$F_{j+1} U = \Pi_F \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} V$$

  with $\|L_{22}\|_2 < \tau \|F_{j+1}\|_2$ and $L_{11} \in \mathbb{R}^{t\times t}$.

  Compress columns of $F_{j+1}U$ to size t:

$$F_{j+1} \leftarrow \Pi_F \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}.$$

  if $t < r$ **then**

    Multiply G_{j+1} from the left by $V_{11} \in \mathbb{R}^{t\times r}$: $G_{j+1} \leftarrow V_{11} G_{j+1}$.

  else

    Multiply F_{j+1} by $V_{11} \in \mathbb{R}^{t\times r}$: $F_{j+1} \leftarrow F_{j+1} V_{11}$.

  end if

  $j = j + 1$

end while

$\tilde{Y} \leftarrow \frac{1}{\sqrt{2}} F_j$, $\tilde{Z} \leftarrow \frac{1}{\sqrt{2}} G_j$

Remark 4.1.1 (Stopping criterion) For stopping the iteration we use the criterion (3.15) from Section 3.1.1:

$$\max\{\|A_j + I_n\|_2, \|B_j + I_m\|_2\} \leq \text{tol}.$$

Since A and B are assumed to be stable, the corresponding iterates tend to $-I$. The stopping criterion is meaningful even in case of formatted arithmetic since the identity is contained in the class of $\mathcal{H}$-matrices. Note that an approximate spectral norm can be computed without much effort since it is observed that in many applications only 5 power iteration steps are needed to compute the norm up to a relative error of 10 percent, see [75, Remark 4.33]. In our examples we compute the approximate spectral norm with 10 power iteration steps. □

Remark 4.1.2 (Scaling) Scaling is introduced to accelerate the initial convergence. In our experiments we use the adapted variant of the "optimal" norm scaling as described in Remark 3.1.7,

$$c_j^{(\text{opt.})} = \left(\frac{\sqrt{\|A_{\mathcal{H},j}^{-1}\|_2 \|B_{\mathcal{H},j}^{-1}\|_2}}{\sqrt{\|A_j\|_2 \|B_j\|_2}} \right)^{\frac{1}{2}}.$$

In the partitioned iteration scheme of Algorithm 4 scaling is incorporated in the following way:

$$\begin{aligned}
A_{j+1} &\leftarrow \frac{1}{2}(c_j A_j \oplus \frac{1}{c_j} A_{\mathcal{H},j}^{-1}), \\
B_{j+1} &\leftarrow \frac{1}{2}(c_j B_j \oplus \frac{1}{c_j} B_{\mathcal{H},j}^{-1}), \\
F_{j+1} &\leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{c_j} F_j, & \frac{1}{\sqrt{c_j}} A_{\mathcal{H},j}^{-1} F_j \end{bmatrix}, \\
G_{j+1} &\leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{c_j} G_j \\ \frac{1}{\sqrt{c_j}} G_j B_{\mathcal{H},j}^{-1} \end{bmatrix}.
\end{aligned}$$

Due to error amplification during the sign function iteration with formatted arithmetic, it is proposed in [75] to scale only in the first iteration step.

As an illustration, the $\mathcal{H}$-matrix based sign function iteration is applied to a stable, symmetric $\mathcal{H}$-matrix $A_{\mathcal{H}}$ of size $n = 4096$ with blockwise accuracy $\epsilon = 10^{-4}$ as computed by (2.30) in Example 2.3.18. We compare different scaling strategies, which are applied either in all or only in the first iteration step until the stopping criterion with $\text{tol} = 10\, n \sqrt{\text{EPS}}$ is satisfied. In Figure 4.1 we observe fastest convergence for the "optimal" norm scaling, followed by the Frobenius norm scaling. The determinantal scaling performs worse, but also reduces the number of steps to reach the stopping criterion compared to the unscaled algorithm. For all choices, the number of iterations compares favorably if scaling is applied in each step instead of the first step only. Nevertheless, the computational time increases significantly if we use scaling for more than one iteration step. This is caused by an enlargement of the blockwise ranks throughout the sign iteration. The maximum rank using the "optimal" norm

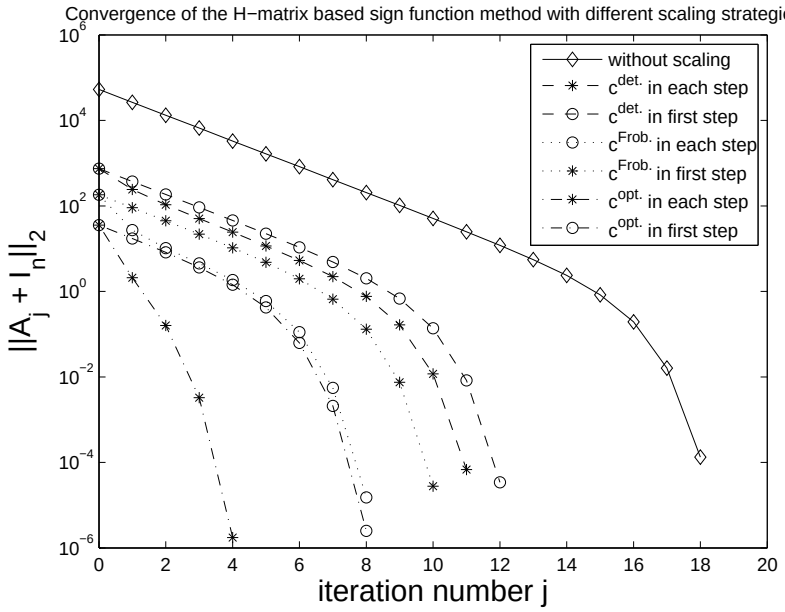


Figure 4.1: Convergence history for $\mathrm{sign}(A_{\mathcal{H}})$ with $n = 4096$.

scaling in each step increases up to 172 whereas it is bounded by 58 if the scaling is applied only in the first step. This can partially be explained by the condition (2.29) for the blockwise rank decision, where the rank k is determined as minimal integer which satisfies the relative criterion

$$\sigma_{k+1} \leq \epsilon\, \sigma_1$$

with singular values $\sigma_1 \geq \sigma_2 \geq \cdots > 0$. Scaling steers the singular values of $A_{\mathcal{H}}$ close to 1 such that the truncation operator $\mathcal{T}_{\mathcal{H},\epsilon}$ (using (2.29)) determines larger blockwise ranks for off-diagonal blocks which are not close to 0. These off-diagonal blocks converge to zero in the course of the iteration with remaining error depending on ϵ. Since the singular values are nearly equal, this error is approximated as Rk-matrix with relatively large rank. To avoid this, an absolute criterion as

$$\sigma_{k+1} \leq \epsilon_{\mathrm{abs}}$$

should be used in the adaptive arithmetic for problems with off-diagonal blocks converging to zero. It is announced that the next release of the HLib includes this feature. We illustrate the observation for a smaller problem size of $n = 1024$ in Table 4.1. We observe an increase of the blockwise ranks as depicted in the green (light grey) during the sign function iteration for the different strategies: without scaling, with "optimal" norm scaling applied either in the first or in all steps. The iteration scheme with scaling in each step reaches the stopping criterion after 5 steps but in larger computational time than the scheme, where scaling is applied once (with 8 steps for convergence). This is explained by the larger blockwise ranks which increase the complexity of the

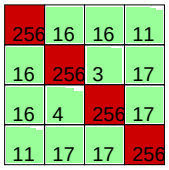
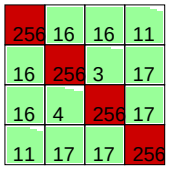
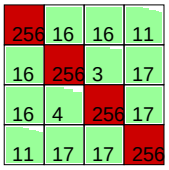
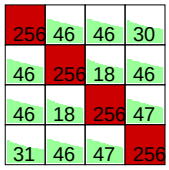
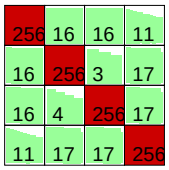
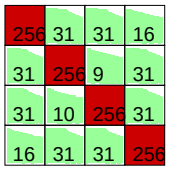
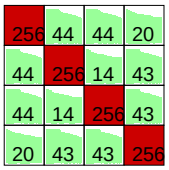
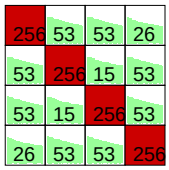
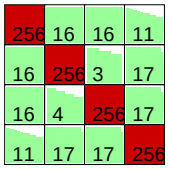
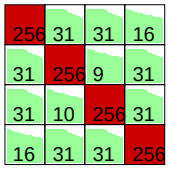
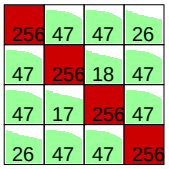
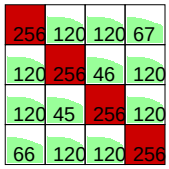
	A_1	A_2	A_3		$A_{\text{conv.}}$	CPU [sec.]
no scaling				...		18 it. 23.37
opt. scaling		in first step 		...		8 it. 13.83
opt. scaling		in all steps 		...		5 it. 17.11

Table 4.1: Convergence of $\mathcal{H}$-matrix based sign function for $n = 1024$.

formatted arithmetic. The singular values of the corresponding iterates A_i are depicted in Figure 4.2 for $i = 1, 2, 3$ and after convergence. To summarize, scaling once is advantageous compared to no scaling for this example since it increases the speed of convergence significantly. Therefore, we use the "optimal" norm scaling for the first iteration step in the numerical experiments throughout this section.

□

Complexity. To apply the algorithm in the large-scale setting we are interested in a reduced computational complexity as well as in a low storage requirement. We reduce the necessary amount of real numbers approximating $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{m\times m}$ in the set of hierarchical matrices $\mathcal{M}_{\mathcal{H},\epsilon}(T_{I\times I})$. As described in Remark 2.3.12 we obtain a bound for the storage requirements of $A_{\mathcal{H}}$ and $B_{\mathcal{H}}$ where the cardinality of the set I is given by n or m, respectively. If k is the maximum rank determined by the blockwise accuracy ϵ, the number of stored real numbers for the matrix $A_{\mathcal{H}}$ is bounded by

$$\mathcal{S}_{\mathcal{H}}(T_{I\times I}, k, n) \leq 2\,\mathrm{C_{sp}}\, n(p+1)k \tag{4.1}$$

instead of n^2. The storage requirements of $B_{\mathcal{H}}$ are bounded analogously. How large the requirement for a typical application actually is, is demonstrated in Table 2.1 in Section 2.3.

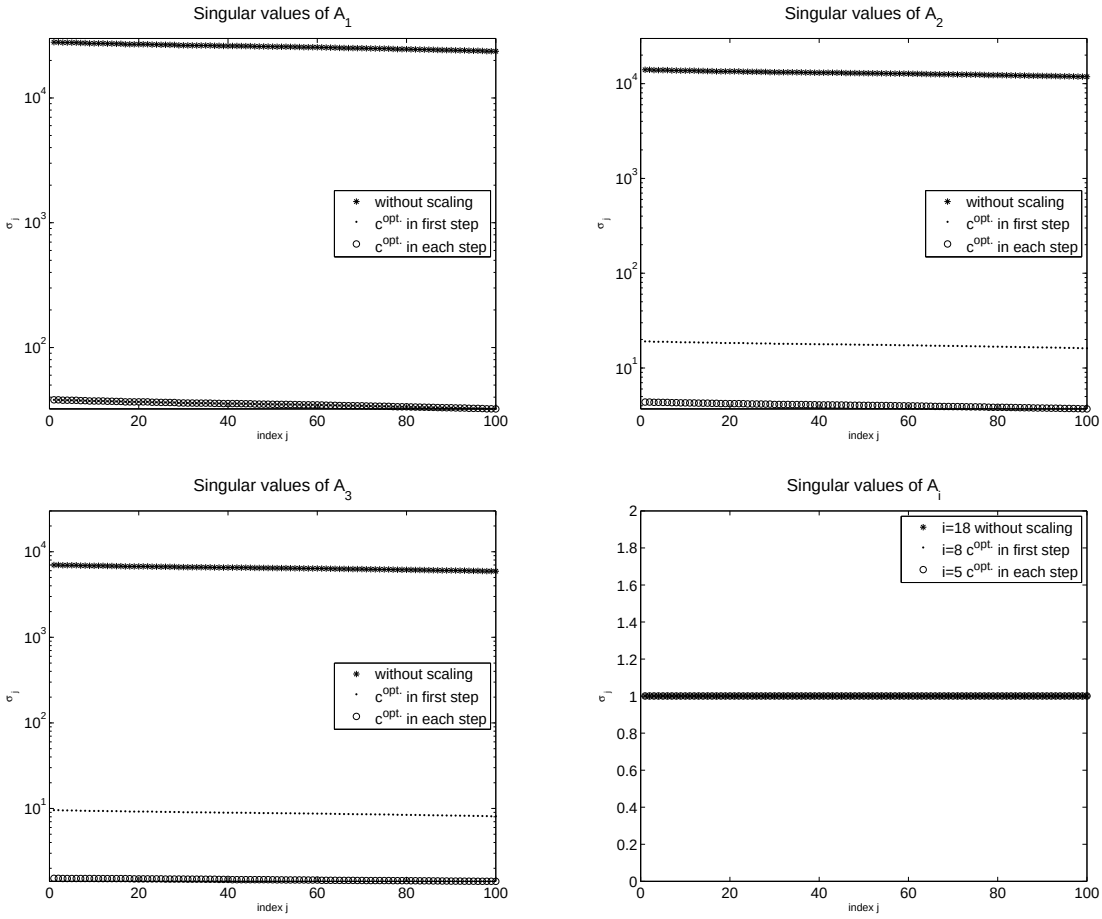


Figure 4.2: Singular values of $A_{\mathcal{H}}$ corresponding to the Figures in 4.1.

With the same assumptions as used in the complexity estimates of Section 2.3, the required workspace for the data-sparse sign function iteration is reduced to about

$$\begin{aligned} \mathcal{S}_{\mathcal{H}-Sign}(T_{I\times I}, k, n, m, X) &\leq 2\,(\mathrm{C_{sp}}\, k\,[n(p+1) + m(p+1)] \\ &\quad + 2(n+m)n_\tau(X)) \qquad (4.2) \\ &= \mathcal{O}((n\log_2(n) + m\log_2(m))k + (n+m)n_\tau(X)) \end{aligned}$$

real numbers. That is, instead of a quadratic demand as needed for the dense sign function iteration in Section 3.1.2, the workspace is reduced to a linear-logarithmic dependency on n and m.

For an estimate of the complexity of Algorithm 4, we consider the parts of the iteration scheme with a reduced number of flops using formatted arithmetic compared to the standard dense arithmetic in (3.17). As seen in Remark 2.3.17, the complexity of computing approximate inverses in line 5 (and 8, replacing n by m) of Algorithm 4 is bounded by

$$\mathcal{N}_{\mathcal{H}-LU}(T_{I\times I}, k, n) = \mathcal{O}(n\log_2^2(n)k^2).$$

For the product of the $\mathcal{H}$-matrix $A_{\mathcal{H},j}^{-1}$ and the dense matrix $F_j \in \mathbb{R}^{n\times r}$ in line 7, we estimate the number of flops by r times the cost of the formatted matrix-vector product

$$\mathcal{N}_{\mathcal{H}\cdot F}(T_{I\times I}, k, n, r) \leq 4r\, \mathrm{C_{sp}}\, n(p+1)k = \mathcal{O}(rn\log_2(n)k).$$

The bound for computing G_{j+1} in line 10 is of the same size with n replaced by m. That is, instead of a cubic complexity, one iteration step in the $\mathcal{H}$-matrix based sign function iteration can be computed in linear-polylogarithmic complexity

$$\mathcal{N}_{\mathcal{H}-Sign}(T_{I\times I}, k, n, m) = \mathcal{O}\left(n\log_2^2(n)k^2 + m\log_2^2(m)k^2\right).$$

For the numerical experiments in this chapter the number of iteration steps is proportional to $\log_2(n)$ (resp. $\log_2(m)$) (see [79, Lemma 3.5] for a general statement). Then, the overall complexity for solving Sylvester equations has a logarithmic dependency of power 3 on n and m.

Remark 4.1.3 Another well known iterative method for the solution of large-scale Lyapunov equations is the low-rank ADI iteration. This method is particularly adapted for sparse problems; when applied to dense systems it is of cubic complexity and therefore of limited use, e.g. for problems as they arise by the finite element discretization of boundary integral equations. Note that the ADI iteration can also be reformulated in formatted arithmetic. Then, in each ADI step an $\mathcal{H}$-LU or $\mathcal{H}$-Cholesky factorization followed by $\mathcal{H}$-forward and backward substitution has to be computed. The complexity of one iteration step is therefore comparable to the number of flops in one iteration step of the sign function iteration. But note that due to the linear or at most superlinear convergence rate ADI would in general require a lot more steps than the quadratically convergent sign function method. □

For Algorithm 4 only A_j and B_j have an impact on the convergence and so the convergence results for the sign function iteration from [75, 79] are valid. If we have convergence in A_j/B_j, further iterations will not improve the iterates F_j and G_j. Therefore we mainly analyze the forward error of the F_j's in the next section.

4.1.1 Error Bounds

We first recall a well known inequality and a subsequent bound which will be used for some results in this section.

Lemma 4.1.4 *If $\|M\|_2 < 1$ for some square matrix $M \in \mathbb{R}^{n\times n}$, then*

$$\|(I-M)^{-1}\|_2 \leq \frac{1}{1-\|M\|_2}. \tag{4.3}$$

Proof: See [74, Lemma 2.3.3]. □

Corollary 4.1.5 *If $M \in \mathbb{R}^{n\times n}$ is nonsingular and its perturbation $\tilde{M} \in \mathbb{R}^{n\times n}$ satisfies*

$$\|I - \tilde{M}M^{-1}\|_2 = \|(M - \tilde{M})M^{-1}\|_2 < 1,$$

then a bound for the norm of the inverse of the perturbation is given by

$$\|\tilde{M}^{-1}\|_2 \leq \|M^{-1}\|_2 \frac{1}{1 - \|M - \tilde{M}\|_2 \|M^{-1}\|_2}. \tag{4.4}$$

Proof: We obtain by simple calculation

$$\tilde{M}^{-1} = M^{-1}[I - (M - \tilde{M})M^{-1}]^{-1}.$$

Invoking inequality (4.3) yields a bound for the inverse $[I - (M - \tilde{M})M^{-1}]^{-1}$ and thus (4.4). □

In this section we derive bounds for the forward error in the perturbed iterates A_j and F_j computed by Algorithm 4, denoted by $\tilde{A}_j$ and $\tilde{F}_j$ throughout this section. The errors in the B- and G-iterates are bounded analogously. All perturbations under consideration stem from using the adaptive arithmetic. In the analysis, we neglect errors introduced by the RRQR/RRLQ factorizations. The derived bounds then suggest a choice of the threshold τ for the numerical rank decision in the RRQR/RRLQ factorization, so that the original bounds also hold approximately for the complete algorithm. To simplify the analysis, the proposed scaling is not taken into account. We consider the following notation and general assumption in analogy with [75, 79].

Notation 4.1.6 *Let j_{max} denote the maximal number of iteration steps. For the perturbed iterates $\tilde{A}_j$ arising in Algorithm 4 we write for the maximum $\mathcal{H}$-matrix inversion error*

$$\delta := \max_{j=0,\ldots,j_{max}} \|\tilde{A}_{\mathcal{H},j}^{-1} - \tilde{A}_j^{-1}\|_2$$

and for the maximum $\mathcal{H}$-matrix addition error

$$\rho := \max_{j=0,\ldots,j_{max}} \|(\tilde{A}_j \oplus \tilde{A}_{\mathcal{H},j}^{-1}) - (\tilde{A}_j + \tilde{A}_{\mathcal{H},j}^{-1})\|_2.$$

Notation 4.1.7 *For the distance between the exact and the perturbed iterates we define*

$$\begin{aligned} \eta_0 &:= \|A_{\mathcal{H}} - A\|_2, \\ \eta_j &:= \|\tilde{A}_j - A_j\|_2, \quad \textit{for all } j = 1, \ldots, j_{max}. \end{aligned}$$

Assumption 4.1.8 *We assume that*

$$\eta_j \, \alpha < 1, \quad \textit{for all } j = 0, \ldots, j_{max}$$

where

$$\alpha := \max_{j=0,\ldots,j_{max}} \|A_j^{-1}\|_2.$$

For the perturbed A-iterates we obtain the following bound which is similar to the bound derived in ([79, Theorem 5.1], [75, Theorem 8.8]) but presented in a slightly modified form.

Proposition 4.1.9 *With Assumption 4.1.8, the forward error of the perturbed iterates in line 6 of Algorithm 4 satisfies for all $j = 0, \ldots, j_{max}$*

$$\eta_{j+1} = \|\tilde{A}_{j+1} - A_{j+1}\|_2 \leq \frac{1}{2}\left(\rho + \delta + \eta_j + \frac{\eta_j \alpha^2}{1 - \eta_j \alpha}\right). \tag{4.5}$$

Proof: By simple calculations, where Assumption 4.1.8 is used in the last inequality, we obtain the following bounds for the forward error in $\tilde{A}_j$:

$$\begin{aligned}
\|\tilde{A}_{j+1} &- A_{j+1}\|_2 \\
&= \frac{1}{2}\|\tilde{A}_j \oplus \tilde{A}_{\mathcal{H},j}^{-1} - (A_j + A_j^{-1})\|_2 \\
&= \frac{1}{2}\|\tilde{A}_j \oplus \tilde{A}_{\mathcal{H},j}^{-1} - (\tilde{A}_j + \tilde{A}_{\mathcal{H},j}^{-1}) + (\tilde{A}_j + \tilde{A}_{\mathcal{H},j}^{-1}) - (A_j + A_j^{-1})\|_2 \\
&\leq \frac{1}{2}\left\{\rho + \|\tilde{A}_j + \tilde{A}_{\mathcal{H},j}^{-1} - (A_j + A_j^{-1})\|_2\right\} \\
&\leq \frac{1}{2}\left\{\rho + \|\tilde{A}_j - A_j + \tilde{A}_{\mathcal{H},j}^{-1} - \tilde{A}_j^{-1} + \tilde{A}_j^{-1} - A_j^{-1}\|_2\right\} \\
&\leq \frac{1}{2}\left\{\rho + \delta + \eta_j + \|\tilde{A}_j^{-1} - A_j^{-1}\|_2\right\} \\
&= \frac{1}{2}\left\{\rho + \delta + \eta_j + \|\tilde{A}_j^{-1}(A_j - \tilde{A}_j)A_j^{-1}\|_2\right\} \\
&\leq \frac{1}{2}\left\{\rho + \delta + \eta_j + \alpha\eta_j\|\tilde{A}_j^{-1}\|_2\right\} \\
&\overset{(4.4)}{\leq} \frac{1}{2}\left\{\rho + \delta + \eta_j + \frac{\eta_j\alpha^2}{1 - \eta_j\alpha}\right\}.
\end{aligned}$$

□

With this proposition it is now possible to derive a bound for the forward error between the exact and the perturbed solution factors.

Theorem 4.1.10 *If Assumption 4.1.8 holds, then the forward error for computing the approximate solution factors $\tilde{F}_j$ is bounded, for all $j = 0, \ldots, j_{max}$, by*

$$\begin{aligned}
&\|\tilde{F}_{j+1} - F_{j+1}\|_2 \\
&\qquad \leq \left(1 + \alpha + \delta + \frac{\eta_j\alpha^2}{1 - \eta_j\alpha}\right)\|\tilde{F}_j - F_j\|_2 + \left(\delta + \frac{\eta_j\alpha^2}{1 - \eta_j\alpha}\right)\|F_j\|_2.
\end{aligned} \tag{4.6}$$

Proof: Using Notation 4.1.6 and 4.1.7 and the definition of α, we get the following bounds:

$$\begin{aligned}
\|\tilde{F}_{j+1} &- F_{j+1}\|_2 \\
&\leq \|\tilde{F}_j - F_j\|_2 + \|\tilde{A}_{\mathcal{H},j}^{-1}\tilde{F}_j - A_j^{-1}F_j\|_2 \\
&\leq \|\tilde{F}_j - F_j\|_2 + \|\tilde{A}_{\mathcal{H},j}^{-1}\tilde{F}_j - \tilde{A}_j^{-1}\tilde{F}_j\|_2 + \|\tilde{A}_j^{-1}\tilde{F}_j - A_j^{-1}F_j\|_2 \\
&\leq \|\tilde{F}_j - F_j\|_2 + \delta\|\tilde{F}_j\|_2 + \|\tilde{A}_j^{-1}\tilde{F}_j - A_j^{-1}\tilde{F}_j\|_2 \\
&\quad + \|A_j^{-1}\tilde{F}_j - A_j^{-1}F_j\|_2 \\
&\leq \|\tilde{F}_j - F_j\|_2 + \delta\|\tilde{F}_j\|_2 + \|\tilde{A}_j^{-1}(A_j - \tilde{A}_j)A_j^{-1}\|_2\,\|\tilde{F}_j\|_2 \\
&\quad + \|\tilde{F}_j - F_j\|_2\|A_j^{-1}\|_2 \\
&\leq (1+\alpha)\|\tilde{F}_j - F_j\|_2 + \delta\|\tilde{F}_j\|_2 + \|\tilde{A}_j^{-1}\|_2\,\eta_j\,\alpha\,\|\tilde{F}_j\|_2.
\end{aligned}$$

If we now use the inequality (4.4) we further obtain

$$\begin{aligned}
\|\tilde{F}_{j+1} &- F_{j+1}\|_2 \\
&\leq (1+\alpha)\|\tilde{F}_j - F_j\|_2 + \delta\|\tilde{F}_j\|_2 + \|A_j^{-1}\|_2\frac{\eta_j\alpha}{1-\eta_j\|A_j^{-1}\|_2}\,\|\tilde{F}_j\|_2 \\
&\leq (1+\alpha)\|\tilde{F}_j - F_j\|_2 + \left(\delta + \frac{\eta_j\alpha^2}{1-\eta_j\alpha}\right)\,\|\tilde{F}_j\|_2 \\
&\leq (1+\alpha)\|\tilde{F}_j - F_j\|_2 + \left(\delta + \frac{\eta_j\alpha^2}{1-\eta_j\alpha}\right)(\|\tilde{F}_j - F_j\|_2 + \|F_j\|_2),
\end{aligned}$$

from which (4.6) follows. □

The analysis shows that we expect an increase of the errors in the F_j's in each step proportional to δ and η_j. Here η_j is again proportional to δ and ρ. Therefore it is sufficient to choose the threshold in the RRQR factorization of the same order as δ and ρ. These bounds for the $\mathcal{H}$-matrix inversion and addition error can be controlled by the adaptive rank choice mentioned at the end of Section 2.3.

The a posteriori error bound (4.6) can be rewritten in a more compact way as follows.

Corollary 4.1.11 *With Assumption 4.1.8 and with*

$$\theta_j := \delta + \frac{\eta_j\alpha^2}{1-\eta_j\alpha},$$

we get the following bound for the forward error, for all $j = 0, \ldots, j_{max}$,

$$\|\tilde{F}_{j+1} - F_{j+1}\|_2 \leq \sum_{\ell=0}^{j}\left(\prod_{i=\ell+1}^{j}(1+\alpha+\theta_i)\right)\theta_\ell\|F_\ell\|_2. \tag{4.7}$$

Proof: We prove (4.7) by induction on j. F_0 is stored in full matrix format, so we get for the initial step:

$$\begin{aligned}\|\tilde{F}_1 - F_1\|_2 &\leq \|\tilde{A}_{\mathcal{H}}^{-1}F - A^{-1}F\|_2 \\ &\leq (\delta + \|\tilde{A}^{-1} - A^{-1}\|_2)\|F\|_2 \\ &\leq (\delta + \frac{\eta_0\alpha^2}{1-\eta_0\alpha})\|F\|_2.\end{aligned}$$

Assuming that (4.7) holds for j,

$$\|\tilde{F}_j - F_j\|_2 \leq \sum_{\ell=0}^{j-1}\left(\prod_{i=\ell+1}^{j-1}(1+\alpha+\theta_i)\right)\theta_\ell\|F_\ell\|_2,$$

and using error bound (4.6), we obtain

$$\begin{aligned}\|\tilde{F}_{j+1} - F_{j+1}\|_2 &\leq (1+\alpha+\theta_j)\,\|\tilde{F}_j - F_j\|_2 + \theta_j\|F_j\|_2 \\ &\leq (1+\alpha+\theta_j)\sum_{\ell=0}^{j-1}\left(\prod_{i=\ell+1}^{j-1}(1+\alpha+\theta_i)\right)\theta_\ell\|F_\ell\|_2 + \theta_j\|F_j\|_2 \\ &= \sum_{\ell=0}^{j-1}\left(\prod_{i=\ell+1}^{j}(1+\alpha+\theta_i)\right)\theta_\ell\|F_\ell\|_2 + \theta_j\|F_j\|_2 \\ &= \sum_{\ell=0}^{j}\left(\prod_{i=\ell+1}^{j}(1+\alpha+\theta_i)\right)\theta_\ell\|F_\ell\|_2.\end{aligned}$$

□

Remark 4.1.12 If Assumption 4.1.8 is strengthened to

$$\exists\lambda \in (0,1)\ \forall j = 0,\ldots,j_{\max}\ :\ \eta_j\,\alpha \leq 1-\lambda,$$

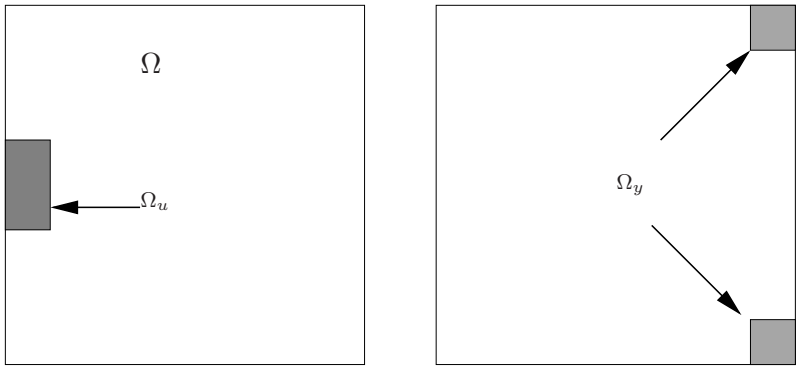
then a straightforward calculation yields that the bound in Proposition 4.1.9 becomes

$$\eta_{j+1} \leq \frac{1}{2}\left(\rho + \delta + \frac{1-\lambda}{\alpha} + \alpha\frac{1-\lambda}{\lambda}\right). \tag{4.8}$$

If α, ρ, $\delta \ll 1$ then η_{j+1} is approximately bounded by $\frac{1}{2}\frac{1-\lambda}{\alpha}$ which is advantageously compared to (4.8). However, in practical circumstances it seems to be unlikely that $\alpha \ll 1$. □

4.1.2 Numerical Results

At the beginning of this section we introduce some general settings which will be needed in most of the numerical simulations of this work.



(a) Control domain. (b) Observation domain.

Figure 4.3: Domain description for the two-dimensional heat equation.

Example 4.1.13 (Main control problem) In our experiments we consider a control problem for the two-dimensional heat equation in a unit square $\Omega = [0,1]^2$ as described in [79]:

$$\begin{aligned} \frac{\partial \mathbf{x}}{\partial t}(t,\xi) &= \Delta \mathbf{x}(t,\xi) + b(\xi)u(t), \qquad \xi \in \Omega,\, t \in (0,\infty), \\ b(\xi) &= \begin{cases} 1, & \xi \in \Omega_u, \\ 0, & \text{otherwise}, \end{cases} \end{aligned}$$

with homogeneous Dirichlet boundary conditions:

$$\mathbf{x}(t,\xi) = 0, \quad \xi \in \partial\Omega, \quad t \in (0,\infty).$$

The subdomain $\Omega_u \subset \Omega$ describes the control domain where we have influence on the system by imposing a heat source $u(\cdot)$. The domain is supposed to be small compared to Ω, see Figure 4.3(a). In the weak form of the partial differential equation we use a classical Galerkin approach with bilinear finite element ansatz functions φ_j on a regular triangulation with $n = (N-1)^2$ inner grid points and mesh size $h = 1/N$. The resulting approximate temperature distribution is given by

$$\mathbf{x}(t,\xi) \;\approx\; \sum_{j=1}^{n} \hat{x}_j(t)\varphi_j(\xi)$$

with n unknowns $\hat{x}_j$, $j = 1,\ldots,n$. The unknowns are restrictions of the temperature $\mathbf{x}(t,\xi)$ to the inner grid points $\xi_{ik} := (ih, kh)$:

$$\hat{x}_j(t) = \mathbf{x}(t,\xi_{ik}), \quad \text{with } j = i + (k-1)(N-1), \quad i,k = 1,\ldots,N-1.$$

We obtain a system of n linear first-order ordinary differential equations which can be written in matrix form as

$$E\dot{\hat{x}}(t) = -\hat{A}\hat{x}(t) + \hat{F}u(t).$$

The matrices $E \in \mathbb{R}^{n\times n}$, $\hat{A} \in \mathbb{R}^{n\times n}$ are given as in Example 2.3.18, $\hat{F} \in \mathbb{R}^n$ is defined by the entries

$$\hat{F}_{i1} = \int_\Omega b(\xi)\varphi_i(\xi)\, d\xi, \quad \text{for } i = 1, \ldots, n.$$

The mass matrix E is symmetric and positive (semi-)definite. With a Cholesky decomposition of $E = LL^T$, we obtain a symmetric, stable system matrix $A = -L^{-1}\hat{A}L^{-T}$ if we multiply the state equation by L^{-1} from the left and define $x := L^T\hat{x}$, and a transformed state equation

$$\dot{x}(t) = Ax(t) + Fu(t), \quad t \geq 0, \tag{4.9}$$

with $F := L^{-1}\hat{F}$. We measure the temperature in Ω_y, see Figure 4.3(b), by setting the entries of the vector $\tilde{G}^T \in \mathbb{R}^n$ corresponding to the grid points $\xi_{ik} \in \Omega_y$ to 1:

$$\tilde{G} = \sum_{\xi_{ik}\in\Omega_y} e^T_{i+(k-1)(N-1)}.$$

This results in an additional output equation

$$y(t) = Gx(t), \quad t \geq 0 \tag{4.10}$$

with $G := \tilde{G}L^{-T}$. Thus, we have an LTI system with a single input and a single output. The system is stable because all eigenvalues of A have negative real part. □

Remark 4.1.14 ($\mathcal{H}$-matrix approximation) For the $\mathcal{H}$-matrix approximation we employ HLib 1.2 [47] which is developed at the Max-Planck-Institute in Leipzig. We use a special $\mathcal{H}$-matrix structure which is shown to be useful for applications based on finite element discretizations of elliptic partial differential operators as for the inversion or the LU-decomposition of the FEM stiffness matrix. In this matrix structure all off-diagonal blocks are stored as Rk-matrices with a blockwise accuracy determined by the parameter ϵ. In the inadmissible leaves along the diagonal, the maximal blocksize is given by $n_{\min} = 256$ throughout all experiments. The parameter in the admissibility condition is chosen as $\eta = 1.0$. To show the usefulness of the data-sparse $\mathcal{H}$-matrix approach, we consider problem sizes with $n > 1000$, otherwise use the standard dense matrix format. □

To limit the additional errors introduced using the formatted arithmetic (see Proposition 4.1.9 and Theorem 4.1.10), it is advised to stop the iteration as soon as possible but not before reaching the domain of quadratic convergence. It is seen that using the stopping criterion (3.15) with a fixed threshold tol = 10^{-4} and performing two additional iteration steps, the prespecified accuracy is fulfilled in all experiments.

As a measure of accuracy we consider the relative residual

$$R(X) = \frac{\|AX + XB + FG\|}{\|A\|\|X\| + \|B\|\|X\| + \|FG\|},$$

which could be considered as the backward error for an approximate solution of the Sylvester equation (up to an amplification factor described in [97, Chapter 15]). With the approximate solution factors $\tilde{Y}$ and $\tilde{Z}$ and by use of the Frobenius norm we compute the residual by help of two QR factorizations,

$$\begin{aligned} \|A\tilde{Y}\tilde{Z} + \tilde{Y}\tilde{Z}B + FG\|_F &= \| \left[A\tilde{Y}, \tilde{Y}, F \right] \left[\tilde{Z}^T, B^T\tilde{Z}^T, G^T \right]^T \|_F \\ &= \|R_1 R_2^T\|_F, \end{aligned}$$

to save storage and computational complexity as proposed in [135, (4.7)] (and implemented in LYAPACK [137]). The two QR factorizations

$$Q_1 R_1 = \left[A\tilde{Y}, \tilde{Y}, F \right], \quad Q_2 R_2 = \left[\tilde{Z}^T, B^T\tilde{Z}^T, G^T \right]$$

are intended to be calculated as "economy-size" factorizations. To compute the residual for large problem sizes, we replace the matrix A by its $\mathcal{H}$-matrix approximant. Numerical tests for smaller problem sizes indicate no significant difference between this "approximate" residual and $R(X)$. For smaller problems the relative error

$$\frac{\|X_* - X\|_F}{\|X_*\|_F}$$

is computed with the reference solution X_* in "full" format and with standard arithmetic.

Remark 4.1.15 All the simulations calculated in this work are performed on an SGI Altix 3700 (32 Itanium II processors, 1300 MHz, 64 GB shared memory, only one processor is used). The algorithms are coded in ANSI C and compiled with the Intel C compiler. We make use of the LAPACK and BLAS libraries from the Intel Math Kernel Library for performing standard dense matrix operations and include the routine DGEQPF of the RRQR library [40] for computing the RRQR factorization. □

Example 4.1.16 In this example we apply the $\mathcal{H}$-matrix based solver to a special variant of Sylvester equation motivated for the purpose of model reduction (see the cross-Gramian approach in Section 5.3 for details),

$$AX + XA + FG = 0. \tag{4.11}$$

The matrices $A \in \mathbb{R}^{n\times n}$ and $F, G^T \in \mathbb{R}^n$ are given by the stable LTI system (4.9) and (4.10). Note that the iteration scheme summarized in Algorithm 4 simplifies since the B-iteration is redundant. This results in a reduced number of flops and a smaller amount of storage demand. The computational time as well as the overall storage requirements are significantly smaller than in the following Examples 4.1.17 and 4.1.18. We vary the problem size from $n = 1024$ to $n = 262,144$ and choose $\tau = 10^{-4}$ for the numerical rank decision in the RRQR factorization and $\epsilon = 10^{-4}$ as approximation error in the adaptive rank choice of the $\mathcal{H}$-matrix arithmetic. With our algorithm we compute the approximate solution factors $\tilde{Y} \in \mathbb{R}^{n\times n_\tau(X)}$ and $\tilde{Z} \in \mathbb{R}^{n_\tau(X)\times n}$ of the so-called

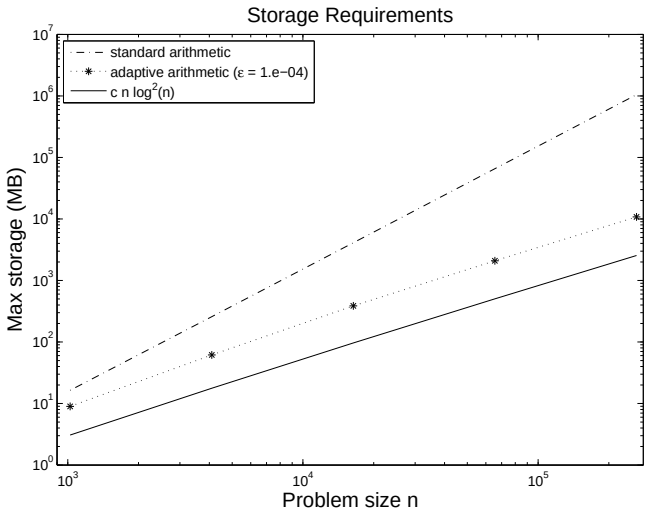


Figure 4.4: Maximal storage requirements in logarithmic scale for Algorithm 4 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line in Example 4.1.16.

cross-Gramian solution $X \in \mathbb{R}^{n \times n}$. We compare the solution from the $\mathcal{H}$-matrix arithmetic based sign function iteration with the solution computed by the original iteration scheme as stated in Section 3.1.2. In the latter scheme all matrices are stored in the usual "full" format and the matrix operations are performed in standard arithmetic. Due to the large memory requirements (see Figure 4.4) these solutions are only computed up to a problem size of $n = 4096$, larger results are extrapolated in the two figures or omitted in Table 4.2. The results of this computation are depicted in columns with column heading "full".

In Figures 4.4 and 4.5 it is seen that the storage requirement as well as the computational time for the algorithm in $\mathcal{H}$-matrix arithmetic exhibits almost linear growth. The ranks of the factors of the cross-Gramian and their accuracy are plotted in Table 4.2. The relative residual is computed up to a problem size of $n = 65,536$ due to storage requirements and seems to be bounded from above for increasing problem size. The proposed algorithm computes approximate low-rank factors $\tilde{Y}$, $\tilde{Z}^T \in \mathbb{R}^{n \times n_\tau(X)}$ which have a very small number of columns, respectively of rows, by a very high accuracy. This demonstrates the efficiency of the developed solver. It should be noted that the largest Sylvester equations solved, one with $n = 262,144$, is equivalent to a linear system of equations with about 68 billion unknowns. For this problem size we get $\tilde{Y}$, $\tilde{Z}^T \in \mathbb{R}^{n \times 18}$ and therefore need 72 MB memory to store the solution instead of 512 GB for the explicit solution X. □

Example 4.1.17 Algorithm 4 is tested in this example with matrices A, B, F, G arising in a semi-discretization of the same control problem in Example 4.1.13. For the space discretization we consider linear finite element ansatz spaces of different sizes n and m which results in different matrix dimensions of the

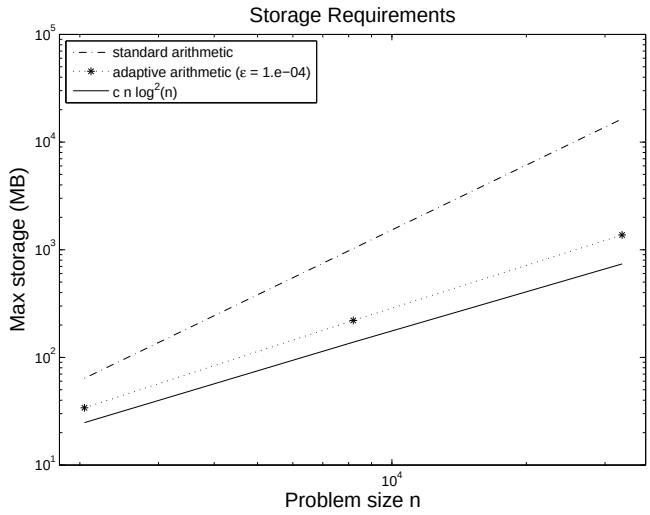


Figure 4.5: CPU time in logarithmic scale for Algorithm 4 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line in Example 4.1.16.

n	# iter.	$n_\tau(X)$		time[sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
1024	11	12	12	19	40	9.7e-08	9.8e-10	2.1e-05
4096	12	13	13	230	2434	7.2e-08	7.0e-10	6.9e-05
16,384	13	15	-	2155	-	2.6e-08	-	-
65,536	14	15	-	15,919	-	1.5e-08	-	-
262,144	15	18	-	131,738	-	-	-	-

Table 4.2: Accuracy and rank $n_\tau(X)$ of the computed solution factors for different problem sizes and $\epsilon = 10^{-4}$, $\tau = 10^{-4}$ in Example 4.1.16.

square matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$ and of $F \in \mathbb{R}^n$, $G^T \in \mathbb{R}^m$. For a fixed size $n = 4096$ we vary the number of grid points m from 1024 to $65,536$. We take the same fixed choice of parameter values, $\epsilon = 10^{-4}$, $\tau = 10^{-4}$, as in Example 4.1.16.

Again, we observe in Table 4.3 high accuracy in the solution factors computed with the algorithm in $\mathcal{H}$-matrix arithmetic by very low numerical ranks $n_\tau(X)$. The relative residual as well as the relative error are observed to remain bounded from above for increasing problem size. The execution time for the algorithm in $\mathcal{H}$-matrix arithmetic is considerably lower than the time needed by the algorithm in standard dense format. □

Example 4.1.18 Now we fix the problem size for the system described in Example 4.1.17 by $n = m = 4096$ and test various parameter combinations of ϵ and τ.

m	# it.	$n_\tau(X)$		time[sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
1024	13	13	13	273	2475	1.2e-07	2.2e-09	7.2e-05
4096	12	13	13	459	4462	7.2e-08	7.0e-10	6.9e-05
16,384	14	14	14	2947	195,070	4.5e-08	8.9e-10	1.5e-04
65,536	15	13	-	53,915	-	4.4e-08	-	-

Table 4.3: Accuracy and rank $n_\tau(X)$ of the computed solution factors for different problem sizes in m and $n = 4096$, $\epsilon = 10^{-4}$, $\tau = 10^{-4}$ in Example 4.1.17.

ϵ	τ	$n_\tau(X)$		time[sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
1.e-04	1.e-04	13	13	459	4462	7.2e-08	7.0e-10	6.9e-05
1.e-06	1.e-04	13	13	917	4462	8.0e-10	7.0e-10	2.8e-07
1.e-08	1.e-04	13	13	1807	4483	7.0e-10	7.0e-10	2.6e-09
1.e-04	1.e-06	24	21	450	4502	7.2e-08	1.0e-13	6.9e-05
1.e-06	1.e-06	21	21	919	4468	3.9e-10	1.0e-13	2.8e-07
1.e-08	1.e-06	21	21	1809	4468	4.0e-12	1.0e-13	2.6e-09
1.e-16	1.e-06	21	21	8919	4464	1.0e-13	1.0e-13	1.4e-14
1.e-04	1.e-08	55	29	457	4491	7.2e-08	4.4e-17	6.9e-05
1.e-06	1.e-08	30	29	899	4492	3.9e-10	4.4e-17	2.8e-07
1.e-08	1.e-08	29	29	1810	4493	4.0e-12	4.4e-17	2.6e-09
1.e-16	1.e-08	29	29	8947	4493	2.5e-17	4.4e-17	1.3e-14
1.e-16	1.e-16	64	70	8948	4518	2.1e-17	3.5e-17	1.2e-14

Table 4.4: Accuracy and rank $n_\tau(X)$ of the computed solution factors for different parameter variations and $n = m = 4096$ in Example 4.1.18.

In all evaluations, 12 iteration steps are needed to reach the stopping criterion. The results of the parameter variation in Table 4.4 show the expected behavior, we have increasing accuracy as ϵ gets smaller. A choice of $\epsilon = 10^{-16}$ results in large computational time and large storage requirements for the A-iterates since the local ranks in the matrix blocks have to be very large to fulfill the accuracy condition from the adaptive rank determination. Therefore the usefulness of the $\mathcal{H}$-matrix approach gets lost. The storage requirements might even get larger than in "full" format if the criterion (2.28) for the local ranks in the Rk-submatrices is not satisfied. It is consequently recommended to choose the parameter ϵ of moderate size. The dimension $n_\tau(X)$ of the solution factors increases with τ getting smaller which has impact on the accuracy for the results in standard arithmetic. We observe that a decrease of τ does not considerably improve the accuracy in the $\mathcal{H}$-matrix computation. The error analysis in Theorem 4.1.10 suggests that no accuracy improvements can be expected by choosing the parameter τ smaller than ϵ. This can be seen in Table 4.4, the relative residual of the $\mathcal{H}$-matrix solution as well as the relative errors are not improved by setting τ smaller or equal to ϵ. It is even seen that for all computations with $\epsilon > \tau$ the numerical rank cannot be predicted correctly which can be explained by the additional errors introduced by larger ϵ in the adaptive arithmetic. In this example it is also observed that for $\epsilon = 10^{-8}$ an increase of τ up to 10^{-6} has no influence on the accuracy. A choice of $\tau = 10^{-4}$ slightly increases the relative residual from 4.0×10^{-12} to 7.0×10^{-10} whereas the relative error is not affected. This observation fits to a criterion presented in [29, page 21] and to the motivation given in Section 3.2.1 for Lyapunov equations, which suggest to choose the RRQR threshold τ of the same order as the square root of the desired accuracy. Since in subsequent computational steps for model order reduction the complexity is mainly determined by $n_\tau(X)$, it is advised to choose τ as large as possible to keep the attainable accuracy with respect to ϵ, thus $\tau \sim \sqrt{\epsilon}$.

□

4.2 $\mathcal{H}$-Matrix Arithmetic Based Sign Function Iteration for Lyapunov Equations

We consider the sign function iteration in the partitioned form (3.25) to compute a low-rank factor $\tilde{Y} \in \mathbb{R}^{n \times n_\tau(X)}$ of the solution $X \in \mathbb{R}^{n \times n}$ of the Lyapunov equation

$$AX + XA^T + BB^T = 0.$$

Since the iteration scheme is one part of the sign function iteration for Sylvester equations we can adapt the data-sparse solver from Section 4.1. We will shortly review the modified scheme with $\mathcal{H}$-matrix format and arithmetic. Earlier results concerning $\mathcal{H}$-based Lyapunov solvers are published in [16, 17].

As in Section 4.1 it is assumed that the matrix $A \in \mathbb{R}^{n \times n}$ can be approximated by a data-sparse $\mathcal{H}$-matrix to initialize the iteration by

$$A_0 = A_{\mathcal{H}}.$$

In the costly part of the iteration the hierarchical matrix arithmetic is incorporated to reduce memory requirements and computational costs:

$$A_{j+1} \leftarrow \frac{1}{2}(A_j \oplus A_{\mathcal{H},j}^{-1}).$$

The formatted inverse is computed by Algorithm 3 using an $\mathcal{H}$-LU decomposition of A_j. This approach has lower storage requirements than computing $A_{\mathcal{H},j}^{-1}$ by a formatted $\mathcal{H}$-matrix inversion. Results of data-sparse Lyapunov solvers with both variants of inversion are published in [17].

The other part of the iteration is stored in the usual "full" format using the formatted matrix-vector product to perform $A_{\mathcal{H},j}^{-1}B_j$,

$$B_{j+1} \leftarrow \frac{1}{\sqrt{2}} \left[\begin{array}{cc} B_j, & A_{\mathcal{H},j}^{-1}B_j \end{array} \right].$$

This part converges to $\tilde{Y} = \frac{1}{\sqrt{2}} \lim_{j\to\infty} B_j$, which is an approximate low-rank factor of X. The increasing number of columns of B_{j+1} again is limited by applying the RRLQ factorization.

Remark 4.2.1 (Stopping criterion) We use the stopping criterion (3.23) for the iteration scheme. With $\lim_{j\to\infty} A_j = -I_n$, using formatted arithmetic, we reach the stopping criterion since the identity is contained in the class of $\mathcal{H}$-matrices. □

Remark 4.2.2 Scaling is used as noted in Remark 3.2.1. Due to error amplification during the sign function iteration with formatted arithmetic, scaling is used only in the first iteration step. □

Complexity. The amount of required workspace is reduced compared to the data-sparse solver for Sylvester equations in (4.2),

$$\mathcal{S}_{\mathcal{H}-Sign}(T_{I\times I}, k, n, X) \leq \mathcal{O}(n\log_2(n)k + nn_\tau(X)),$$

assuming that the same assumptions on the underlying hierarchical structure are valid.

The computational complexity of Algorithm 5 can be bounded to about

$$\mathcal{N}_{\mathcal{H}-Sign}(T_{I\times I}, k, n) = \mathcal{O}\left(n\log_2^2(n)k^2\right)$$

flops for one iteration step. That is, the data-sparse solver for Lyapunov equations has almost linear storage requirements and a linear-polylogarithmic complexity.

4.2.1 Error Bounds

Using the formatted arithmetic introduces further errors besides the usual rounding errors. The influence of these errors in Algorithm 5 on the accuracy of the computed solution factor can be analyzed similar to Section 4.1.1.

Algorithm 5 Calculate low-rank factor $\tilde{Y}$ of X for $AX + XA^T + BB^T = 0$.

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, tol, ϵ, τ

OUTPUT: Approximate low-rank factor $\tilde{Y}$ of the solution X.

1: $A_0 \leftarrow (A)_{\mathcal{H}}$
2: $B_0 \leftarrow B$
3: $j = 0$
4: **while** $\|A_j + I_n\| >$ tol **do**
5: Compute the approximate inverse $A^{-1}_{\mathcal{H},j}$ by Algorithm 3.
6: $A_{j+1} \leftarrow \frac{1}{2}(A_j \oplus A^{-1}_{\mathcal{H},j})$
7: $B_{j+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} B_j, & A^{-1}_{\mathcal{H},j} B_j \end{bmatrix}$
8: Compute an RRLQ factorization

$$B_{j+1} = \Pi L\, Q = \Pi \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} Q$$

with $\|L_{22}\|_2 < \tau \|B_{j+1}\|_2$ and $L_{11} \in \mathbb{R}^{r\times r}$.
9: Compress columns of B_{j+1} to size r:

$$B_{j+1} \leftarrow \Pi \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}.$$

10: $j = j + 1$
11: **end while**
12: $\tilde{Y} \leftarrow \frac{1}{\sqrt{2}} B_{j+1}$

For the forward error in the A-iterates, the result (4.5) from Proposition 4.1.9 derived for Sylvester equations is valid. Furthermore, the distance between B_j and the perturbed $\tilde{B}_j$ can be bounded as in (4.6) in Theorem 4.1.10. Exploiting the symmetry of the solution of a Lyapunov equation we obtain a bound for the relative forward error in the B-iteration.

Corollary 4.2.3 *With the Notation 4.1.6, 4.1.7 and Assumption 4.1.8 from Section 4.1.1 and with*

$$\theta_j := \delta + \frac{\eta_j \alpha^2}{1 - \eta_j \alpha},$$

we obtain a bound for the relative error for all $j = 0, \ldots, j_{max}$:

$$\frac{\|\tilde{B}_{j+1} - B_{j+1}\|_2}{\|B_{j+1}\|_2} \leq \sum_{\ell=0}^{j} \theta_\ell \prod_{i=\ell+1}^{j} (1 + \alpha + \theta_i). \tag{4.12}$$

Proof: To show that (4.12) holds, use the boundedness of the iterates

$$\|B_{j-1}\|_2 \leq \|B_j\|_2 \leq \sqrt{2\sigma_1}, \quad \text{for all } j = 0, \ldots, j_{\max},$$

where σ_1 is the largest eigenvalue of the computed solution $2X = \lim_{j\to\infty} B_j B_j^T$ [74, Corollary 8.6.3]. Then we can derive a bound for the relative error in B_{j+1}:

$$\|\tilde{B}_{j+1} - B_{j+1}\|_2 \leq \left[\sum_{\ell=0}^{j} \theta_\ell \prod_{i=\ell+1}^{j} (1 + \alpha + \theta_i)\right] \|B_{j+1}\|_2.$$

□

Remark 4.2.4 Once again, if the Assumption 4.1.8 is strengthened to

$$\exists \lambda \in (0,1)\ \forall j = 0, \ldots, j_{\max}\ :\ \eta_j\, \alpha \leq 1 - \lambda,$$

and setting

$$\theta := \delta + \alpha \frac{1-\lambda}{\lambda},$$

the relative error (4.12) in Corollary 4.2.3 can be bounded by

$$\frac{\|\tilde{B}_{j+1} - B_{j+1}\|_2}{\|B_{j+1}\|_2} \leq \theta \frac{(1+\alpha+\theta)^{j+1} - 1}{\alpha + \theta}.$$

□

4.2.2 Numerical Results

As in Example 4.1.13, we consider a control problem for the two-dimensional heat equation where we include a diffusion coefficient a:

$$\frac{\partial \mathbf{x}}{\partial t}(t,\xi) = \nabla(a(\xi) \cdot \nabla \mathbf{x}(t,\xi)) + b(\xi)u(t). \tag{4.13}$$

In this setting, a is a material-specific quantity depending on the heat conductivity, the density and the heat capacity. We will set the diffusion coefficient constant in most of the experiments except for Example 4.2.6 where we vary a over the domain Ω. In Example 4.2.7 we include a constant convective term in (4.13) and examine the influence of convection on the algorithm by choosing different constant diffusion coefficients $a \in (0, 1]$. After discretization by linear finite elements we denote the resulting state space form of order n by

$$\dot{x}(t) = Ax(t) + Bu(t).$$

For the $\mathcal{H}$-matrix approximation we use the setting as described in Remark 4.1.14. We test various parameter combinations of ϵ for the blockwise accuracy and τ as threshold for the numerical rank decision in the RRQR factorization. The error analysis in Corollary 4.2.3 suggests $\tau \sim \epsilon$ to obtain

$$\frac{\|Y - \tilde{Y}\|_2}{\|Y\|_2} \sim \epsilon,$$

but we also test other parameter combinations in order to support this analysis by numerical evidence. Recall that the analysis in Section 3.2.2 suggests to

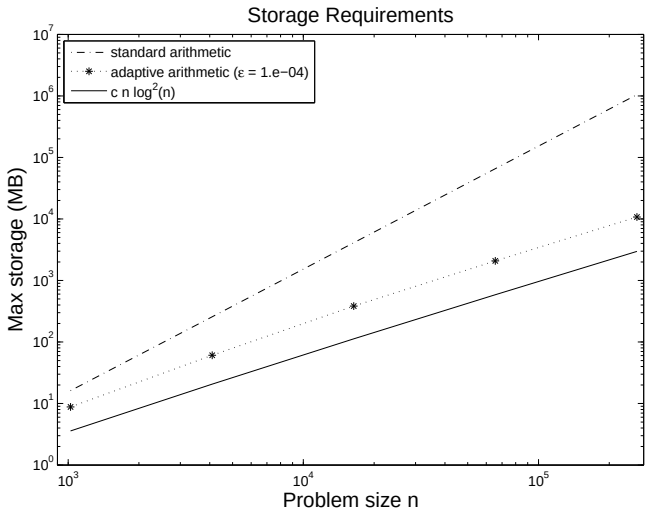


Figure 4.6: Maximum storage requirements in logarithmic scale for Algorithm 5 in $\mathcal{H}$-matrix arithmetic and standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line in Example 4.2.5.

choose τ in the size of the square root of the approximation error, i.e. $\tau \sim \sqrt{\epsilon}$ to compute X up to a relative accuracy of ϵ

$$\frac{\|X - \tilde{Y}\tilde{Y}^T\|_2}{\|X\|_2} \sim \epsilon.$$

For the stopping criterion we choose tol $= 10^{-4}$ and perform two additional iteration steps thereby exploiting the quadratic convergence rate of the sign function iteration.

The relative residual

$$\frac{\|A\tilde{Y}\tilde{Y}^T + \tilde{Y}\tilde{Y}^T A^T + BB^T\|}{2\|A\|\,\|\tilde{Y}\tilde{Y}^T\| + \|B\|^2}$$

is computed by help of two "economy-size" QR factorizations, using the Frobenius norm, as introduced for the Sylvester equation in Section 4.1.2.

Example 4.2.5 First we consider a constant diffusion coefficient: $a(\cdot) \equiv 1.0$. In Figures 4.6 and 4.7 the storage requirements and the computing time of the new solver with $\mathcal{H}$-matrix arithmetic with a fixed choice of $\epsilon = 10^{-4}$ and $\tau = 10^{-4}$ is compared with the sign function method in standard arithmetic. As in Example 4.1.16 we have extrapolated the results of the standard arithmetic computation for sizes larger than $n = 4096$ due to the large memory requirements.

It is observed that the storage requirements as well as the computing time for the algorithms in $\mathcal{H}$-matrix arithmetic exhibit an almost linear complexity. The number of flops as well as the required storage can further be reduced for

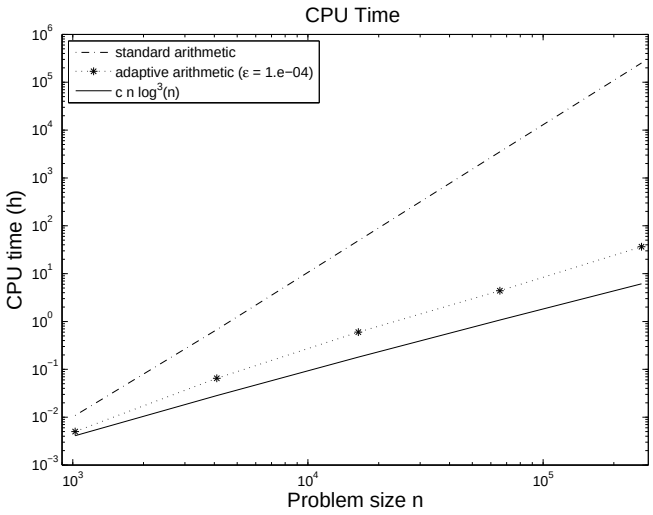


Figure 4.7: CPU time in logarithmic scale for Algorithm 5 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line in Example 4.2.5.

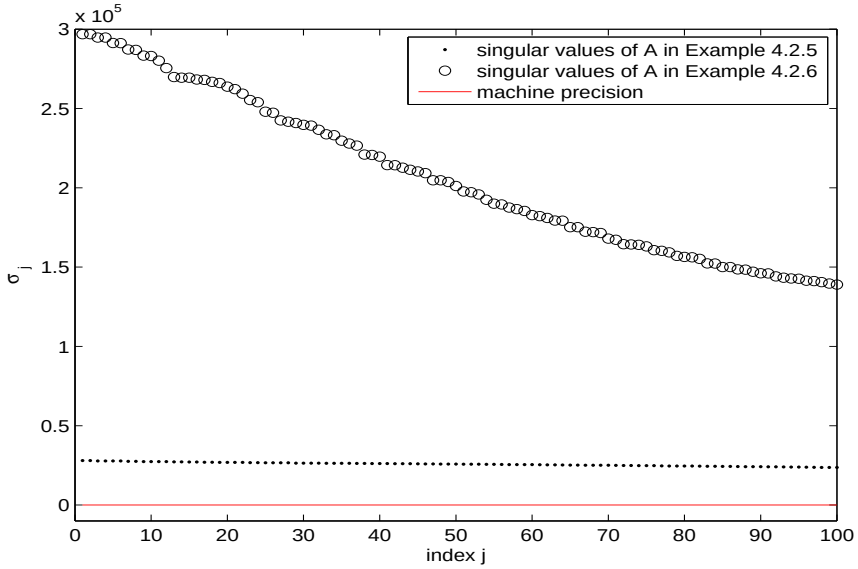


Figure 4.8: Singular values of A for $n = 1024$ and $\epsilon = 10^{-4}$.

this example if the symmetry of the A-iterates is exploited by only storing the upper triangular parts of the iterates. Note that this available structure is not further exploited throughout this work.

Table 4.5 reports the accuracy of the solutions with a problem size n varying from 1024 to 262,144. The relative residual seems to be bounded from

ϵ	τ	# it.	$n_\tau(X)$		time [sec]		rel. residual		rel. error
			$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
$n = 1024$									
1.e-04	1.e-04	11	12	12	18	39	1.3e-07	6.2e-10	3.1e-05
1.e-06	1.e-04	11	12	12	31	39	1.2e-09	6.2e-10	2.5e-07
1.e-08	1.e-04	11	12	12	51	39	6.2e-10	6.2e-10	1.2e-09
1.e-04	1.e-06	11	20	19	19	39	1.3e-07	1.3e-13	3.1e-05
1.e-06	1.e-06	11	19	19	31	39	1.0e-09	1.3e-13	2.5e-07
1.e-08	1.e-06	11	19	19	51	39	7.6e-12	1.3e-13	1.2e-09
1.e-04	1.e-08	11	43	26	19	39	1.3e-07	4.4e-17	3.1e-05
1.e-06	1.e-08	11	26	26	31	39	1.0e-09	4.4e-17	2.5e-07
1.e-08	1.e-08	11	26	26	51	39	7.6e-12	4.4e-17	1.2e-09
$n = 4096$									
1.e-04	1.e-04	12	13	13	234	2436	7.7e-08	6.2e-10	1.4e-04
1.e-06	1.e-04	12	13	13	464	2432	7.2e-10	6.2e-10	3.0e-07
1.e-08	1.e-04	12	13	13	910	2430	6.2e-10	6.2e-10	4.2e-09
1.e-04	1.e-06	12	26	21	235	2432	7.7e-08	4.3e-14	1.4e-04
1.e-06	1.e-06	12	21	21	465	2433	3.6e-10	4.3e-14	3.0e-07
1.e-08	1.e-06	12	21	21	911	2433	4.0e-12	4.3e-14	4.2e-09
1.e-04	1.e-08	12	56	29	238	2438	7.7e-08	3.9e-17	1.4e-04
1.e-06	1.e-08	12	30	29	466	2441	3.6e-10	3.9e-17	3.0e-07
1.e-08	1.e-08	12	29	29	912	2438	4.0e-12	3.9e-17	4.2e-09
$n = 16,384$									
1.e-04	1.e-04	13	16	-	2158	-	2.3e-08	-	-
$n = 65,536$									
1.e-04	1.e-04	14	16	-	15,781	-	1.1e-08	-	-
$n = 262,144$									
1.e-04	1.e-04	15	18	-	131,192	-	4.0e-09	-	-

Table 4.5: Accuracy and rank $n_\tau(X)$ of the computed solution factors from Algorithm 5 for different problem sizes and parameter combinations in Example 4.2.5.

above for increasing problem size. For the parameter variation we observe a similar behavior as in Example 4.1.18 for the solution of Sylvester equations and as expected from the error analysis in Section 4.2.1 and from the analysis in Section 3.2.2. □

Example 4.2.6 In Table 4.6 we report results for Algorithm 5 applied to a modification of the original heat equation example. We vary the diffusion coefficient a in (4.13) over the domain as illustrated in Figure 4.9 and similarly as in [79]:

$$a(\xi) = \begin{cases} 10, & \xi \in [-1,1] \times [-\frac{1}{3}, \frac{1}{3}], \\ 10^{-4}, & \xi \in [-\frac{1}{3}, \frac{1}{3}] \times \left([-1, -\frac{1}{3}) \cup (\frac{1}{3}, 1]\right), \\ 1.0, & \text{otherwise.} \end{cases}$$

In these experiments we fix $n = 4096$ and vary the parameter ϵ and τ as in Example 4.2.5. Since the sign function iteration where scaling is applied only

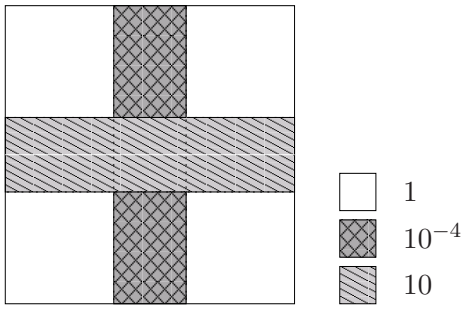


Figure 4.9: Varying diffusion coefficient a in (4.13).

in the first step needs 18 iteration steps to reach the stopping criterion, we use scaling also for the second iteration step. In Figure 4.8 it is seen that the first 100 singular values of A with varying diffusion decrease significantly faster than the singular values of A in Example 4.2.5. To steer the singular values closer to 1, and thus obtain faster convergence, it is advised to scale twice. The resulting iteration needs 13 iteration steps for convergence and a reduced computational time, even though the blockwise ranks of the iterates are increased compared to using scaling only in the first iteration step, see the discussion in Remark 4.1.2. Note that it would be possible to employ an heuristic criterion for the number of scaled iteration steps based on the condition number of A.

In Table 4.6 we observe larger ranks $n_\tau(X)$ of the computed solution factors compared to the ranks of the results for $a(\cdot) \equiv 1.0$ in Table 4.5. This is explained by the smoother decay of the eigenvalues of X for varying diffusion, see Figure 4.10. The relative residuals are nearly of the same size for the standard dense implementation and slightly larger for the data-sparse computation compared to the corresponding results in Example 4.2.5. We observe larger relative errors compared to the corresponding rows in Table 4.5. These larger errors are due to the increasing ill-condition of the Lyapunov equation. An estimate for the 2-norm condition number of the Lyapunov operator yields $6.1 \cdot 10^7$ as compared to $5.3 \cdot 10^3$ for the Lyapunov equation corresponding to $a(\cdot) \equiv 1.0$. For details on the condition number of Lyapunov equations see [97, Chapter 15]. The computational time in all $\mathcal{H}$-matrix based calculations is decreased compared to the time in Table 4.5 for constant diffusion, caused by smaller values for the maximum blockwise rank in the adaptive arithmetic. These lower ranks are due to the fast decay of the singular values of A. Note that the computational time needed for each iteration step in the standard dense implementation is of comparable size for both examples.

ϵ	τ	$n_\tau(X)$		time[sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
1.e-04	1.e-04	24	25	193	2612	3.6e-07	2.5e-10	8.16e-03
1.e-06	1.e-04	25	25	379	2608	2.1e-09	2.5e-10	2.53e-05
1.e-08	1.e-04	25	25	691	2616	2.5e-10	2.5e-10	2.30e-07
1.e-04	1.e-06	41	38	195	2630	3.6e-07	3.7e-14	8.16e-03
1.e-06	1.e-06	39	38	380	2618	2.0e-09	3.7e-14	2.53e-05
1.e-08	1.e-06	39	38	692	2632	1.4e-11	3.7e-14	2.30e-07
1.e-04	1.e-08	87	54	218	2625	3.6e-07	9.2e-17	8.16e-03
1.e-06	1.e-08	54	54	382	2625	2.0e-09	9.2e-17	2.53e-05
1.e-08	1.e-08	54	54	694	2642	1.4e-11	9.2e-17	2.30e-07

Table 4.6: Accuracy and rank $n_\tau(X)$ of the computed solution factors from Algorithm 5 for $n = 4096$ and varying diffusion coefficient in Example 4.2.6.

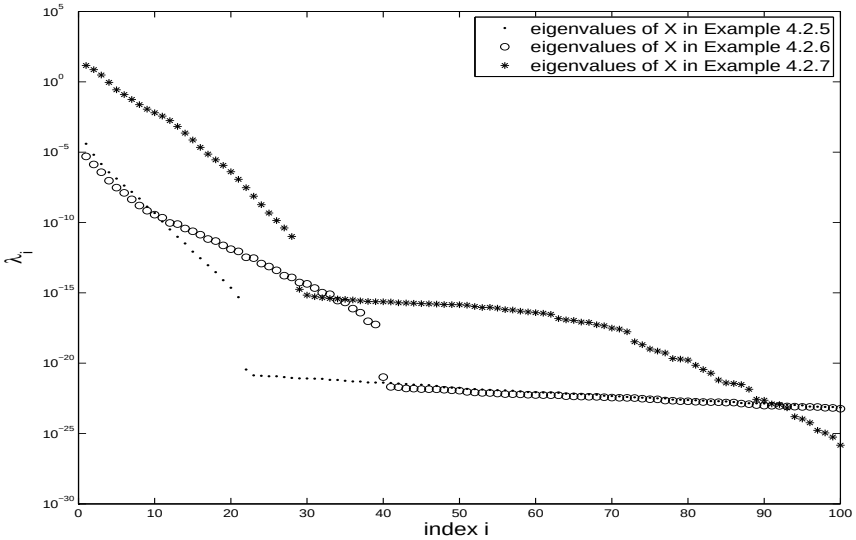


Figure 4.10: Largest 100 eigenvalues of X for $n = 4096$ and $\tau = 10^{-6}$.

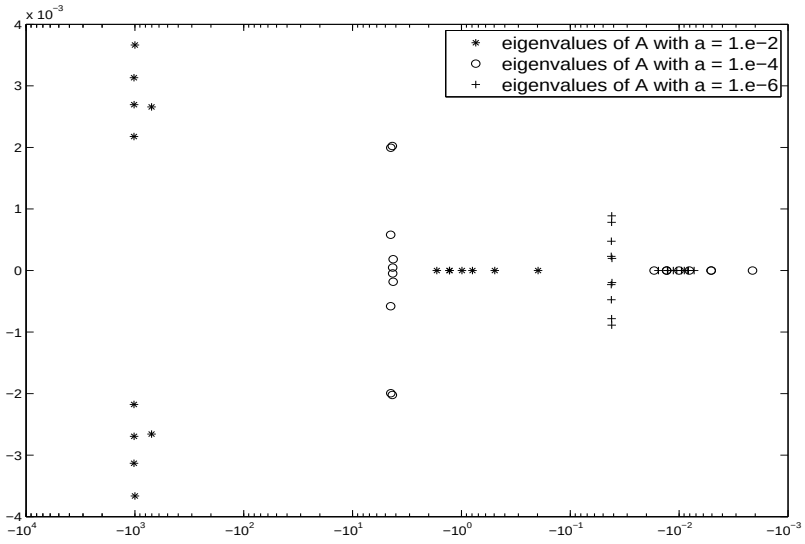


Figure 4.11: Eigenvalues of A for $n = 4096$ with convection.

□

Example 4.2.7 Next we add constant convection to the heat equation (4.13). Thus, we apply Algorithm 5 to a system with nonsymmetric stiffness matrix A resulting from the convection-diffusion equation

$$\frac{\partial \mathbf{x}}{\partial t}(t,\xi) = a\,\Delta \mathbf{x}(t,\xi) + c \cdot \nabla \mathbf{x}(t,\xi) + b(\xi)u(t), \qquad \xi \in \Omega,\ t \in (0,\infty),$$

with a constant diffusion coefficient $a \in (0,1]$ and a fixed choice of the convection vector $c = (0,1)^T$. We choose different values for a to analyze the influence of the convective term. For a fixed problem size $n = 4096$ we depict the critical part of the spectrum of A for the diffusion coefficients $a = 10^{-2}$, $a = 10^{-4}$ and $a = 10^{-4}$ in Figure 4.11. The eigenvalue closest to the imaginary axis is $\lambda \approx 2. \times 10^{-3}$ for the diffusion coefficient $a = 10^{-4}$. As already discussed in Section 3.1.1, the Newton iteration suffers from numerical problems if the distance of the closest eigenvalue is smaller than the square of ϵ. Nevertheless, we observe that the Newton iteration converges even using a relatively large value of $\epsilon = 10^{-4}$ for the adaptive arithmetic.

Note that the partitioning of the underlying $\mathcal{H}$-tree as well as the admissibility condition are not particularly adapted for the case of dominant convection. For modifications to reduce the approximation error between the original matrix and its $\mathcal{H}$-matrix approximant for problems with convection see [49]. Nevertheless, it is shown in [51] that the $\mathcal{H}$-LU-factorization works well also in the convection-dominant case without any modification on the index clustering. All calculations for $a > 10^{-6}$ fulfill the stopping criterion after 12 iteration steps; for $a = 10^{-6}$ two steps less are needed. Since the rank is chosen adaptively, the

ϵ	τ	$n_\tau(X)$		time[sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
$a = 0.01$								
1.e-04	1.e-04	13	13	228	2434	7.6e-08	6.2e-10	1.35e-04
1.e-06	1.e-04	13	13	454	2433	7.2e-10	6.2e-10	2.98e-07
1.e-08	1.e-04	13	13	882	2429	6.2e-10	6.2e-10	4.39e-09
1.e-04	1.e-06	26	21	229	2432	7.6e-08	4.3e-14	1.35e-04
1.e-06	1.e-06	21	21	455	2432	3.6e-10	4.3e-14	2.98e-07
1.e-08	1.e-06	21	21	882	2432	4.1e-12	4.3e-14	4.24e-09
1.e-04	1.e-08	59	29	232	2436	7.6e-08	4.3e-17	1.35e-04
1.e-06	1.e-08	30	29	455	2437	3.6e-10	4.3e-17	2.98e-07
1.e-08	1.e-08	29	29	883	2440	4.1e-12	4.3e-17	4.24e-09
$a = 10^{-4}$								
1.e-04	1.e-04	14	14	230	2433	7.3e-08	1.2e-10	1.29e-04
1.e-06	1.e-04	14	14	455	2425	3.9e-10	1.2e-10	2.83e-07
1.e-08	1.e-04	14	14	884	2430	1.2e-10	1.2e-10	3.89e-09
1.e-04	1.e-06	26	21	232	2433	7.1e-08	3.0e-14	1.28e-04
1.e-06	1.e-06	21	21	457	2429	3.7e-10	3.0e-14	2.83e-07
1.e-08	1.e-06	21	21	885	2435	3.6e-12	3.0e-14	3.77e-09
1.e-04	1.e-08	62	29	235	2437	7.1e-08	3.5e-17	1.28e-04
1.e-06	1.e-08	31	29	457	2420	3.7e-10	3.5e-17	2.83e-07
1.e-08	1.e-08	29	29	885	2433	3.6e-12	3.5e-17	3.77e-09
$a = 10^{-6}$								
1.e-04	1.e-04	21	21	736	2062	4.5e-08	4.7e-11	1.42e-05
1.e-06	1.e-04	21	21	1497	2059	4.6e-10	4.7e-11	1.12e-07
1.e-08	1.e-04	21	21	2086	2058	4.7e-11	4.7e-11	7.40e-10
1.e-04	1.e-06	29	28	738	2054	4.5e-08	5.7e-15	1.42e-05
1.e-06	1.e-06	28	28	1503	2061	4.6e-10	5.7e-15	1.12e-07
1.e-08	1.e-06	28	28	2083	2054	3.5e-12	5.7e-15	7.29e-10
1.e-04	1.e-08	46	33	741	2055	4.5e-08	1.8e-17	1.42e-05
1.e-06	1.e-08	33	33	1505	2052	4.6e-10	1.8e-17	1.12e-07
1.e-08	1.e-08	33	33	2087	2065	3.5e-12	1.8e-17	7.29e-10

Table 4.7: Accuracy and rank $n_\tau(X)$ of the computed solution factors from Algorithm 5 for $n = 4096$ with convective term and different values of the diffusion coefficient a in Example 4.2.7.

accuracy results in Table 4.7 are nearly the same for all diffusion coefficients and compare well with the results in Table 4.5 for problems without convection. For $a \to 0$ we observe an increase in the ranks in the $\mathcal{H}$-matrix approximation due to the declined convergence of the $\mathcal{H}$-matrix approximant towards the original matrix in convection-dominant problems [49]. This results in larger computational time for the data-sparse solver. The numerical ranks of the computed solutions X are comparable small to the ranks in Example 4.2.5 without convection. This can be explained by the steep decay of the eigenvalues of X, the largest 100 real eigenvalues are depicted in Figure 4.10 (for $a = 10^{-6}$).

□

Example 4.2.8 In this example we consider a boundary element discretization of the Laplace equation in $\Omega \subset \mathbb{R}^3$. Using the Ritz-Galerkin method with n piecewise constant ansatz functions $\{\varphi_1, \ldots, \varphi_n\}$ we obtain the following entries of the stiffness matrix

$$A_{ij} = \int_\Gamma \varphi_i(y) \int_\Gamma \frac{1}{4\pi} \frac{1}{\|x-y\|_2} \varphi_j(x)\, d\Gamma_x d\Gamma_y$$

for $i, j = 1, \ldots, n$, see [88] for details. To construct a dynamical system we introduce an artifical time dependence. By use of the stiffness matrix A, taking $B \in \mathbb{R}^{n\times 1}$ as in Example 4.1.13, we obtain a state equation in standard form with stable matrix A. We choose Ω as a three-dimensional sphere and compute the entries in the low-rank blocks of the $\mathcal{H}$-matrix using adaptive cross approximation [45] with a blockwise accuracy of 10^{-7}. From the resulting $\mathcal{H}$-matrix, a coarser approximation with relative error of 10^{-6} is computed. The Lyapunov equation for the solution of the controllability Gramian is solved by the data-sparse Algorithm 5 using a fixed parameter choice of $\epsilon = \tau = 10^{-4}$.

In Table 4.8 we observe a good accuracy for the computed low-rank solution factors. The computing time is larger compared to similar problem sizes of the sparse problems, but note that for $n = 32,768$ a system of about 1 billion unknowns results. The solution of a problem of this size, involving dense matrices, is only possible by use of a data-sparse solver with formatted arithmetic. The maximum storage requirements throughout the sign function iteration are also reduced significantly, see Figure 4.12. Furthermore, computing low-rank factors $\tilde{Y} \in \mathbb{R}^{n\times 12}$ reduces the amount of storage for the solution from 8 GB to 3 MB for $n = 32,768$.

n	# it.	$n_\tau(X)$		time [sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
2048	11	10	10	228	282	9.4e-07	9.4e-07	2.1e-06
8192	11	11	11	1769	17,071	4.6e-07	4.6e-07	2.6e-06
32,768	12	12	-	15,306	-	7.7e-07	-	-

Table 4.8: Accuracy and rank $n_\tau(X)$ of the computed solution factors from Algorithm 5 in Example 4.2.8.

□

4.3 Extension to Generalized Lyapunov Equations

In this section, we show how the derived results can be extended to generalized Lyapunov equations of the form (3.27),

$$AXE^T + EXA^T + BB^T = 0,$$

where $A, E \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. To initialize the iteration, the matrices E and A are approximated in the $\mathcal{H}$-matrix format. Since we cannot say anything

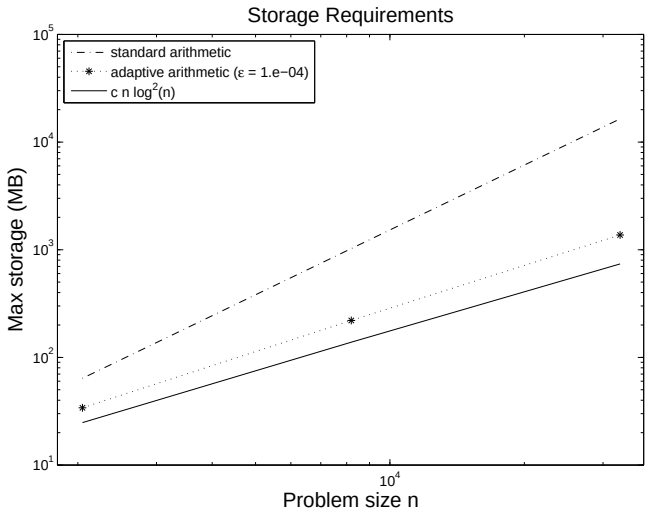


Figure 4.12: Maximum storage requirements in logarithmic scale for Example 4.2.8 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line.

about the growth of the blockwise ranks during the iteration we avoid the computation and storage of EA_j^{-1} and $A_j^{-1}E$ in (3.30a) computing an $\mathcal{H}$-LU-factorization of A_j instead. The matrix product $EA_j^{-1}E$ in (3.30a) is computed by

$$[L_{\mathcal{H}}, U_{\mathcal{H}}] \leftarrow LU_{\mathcal{H}}(A_j),$$

followed by an $\mathcal{H}$-forward and $\mathcal{H}$-backward substitution for the solution of $L_{\mathcal{H}}^{-1}E$ and $EU_{\mathcal{H}}^{-1}$,

$$\begin{aligned} L_{\mathcal{H}} W &= E, \\ V U_{\mathcal{H}} &= E. \end{aligned}$$

By formatted multiplication we obtain

$$EA_j^{-1}E \approx V \odot W.$$

Remark 4.3.1 If E is symmetric and positive definite and A is symmetric and negative definite then all iterates A_j are also symmetric and negative definite in exact arithmetic. In this case we replace the $\mathcal{H}$-LU-decomposition by the Cholesky decomposition of $-A_j$ in approximate arithmetic, see [120] for details. Note that the Cholesky decomposition is performed with half of the complexity. □

The update of the B_j-iterates is performed by the following two steps (replacing (3.30b) in the standard dense implementation),

$$\begin{aligned} L_{\mathcal{H}} \tilde{B}_j &\leftarrow B_j, \\ B_{j+1} &\leftarrow \frac{1}{\sqrt{2}} \left[\; B_j, \;\; V\tilde{B}_j \; \right], \end{aligned}$$

where a formatted triangular solver is used to compute $\tilde{B}_j$.

Remark 4.3.2 (Stopping criterion) Since $\lim_{j\to\infty} A_j = -E$ and E is contained in the set of $\mathcal{H}$-matrices, we use

$$\|A_j + E\|_2 \leq \text{tol} \cdot \|E\|_2$$

as stopping criterion for the iteration as proposed in Section 3.2.3. □

We obtain

$$\tilde{Y} = \frac{1}{\sqrt{2}} E_{\mathcal{H}}^{-1} \lim_{j\to\infty} B_j$$

as a factor of the approximate solution $X \approx \tilde{Y}\tilde{Y}^T$ of (3.27). The inversion of E in this solution formula is performed by a formatted LU-decomposition, see Definition 2.3.17.

Remark 4.3.3 (Scaling) As motivated in Section 3.2.3, the statements in the inner loop of Algorithm 6 are replaced by

$$\begin{aligned} A_1 &\leftarrow \frac{1}{2}(cA_0 \oplus \frac{1}{c} V \odot W), \\ B_1 &\leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{c}B_0 & \frac{1}{\sqrt{c}} V B_0 \end{bmatrix}, \end{aligned}$$

in the first iteration step, using

$$c = \sqrt{\frac{\|E A_{\mathcal{H},j}^{-1} E\|_2}{\|A_j\|_2}},$$

in order to accelerate convergence. □

The hierarchical matrix format and the approximate arithmetic is introduced in the iteration scheme as summarized in Algorithm 6.

Complexity. The storage requirements for the data-sparse generalized Newton iteration are in the size of the data-sparse standard Lyapunov solver plus the storage for the $\mathcal{H}$-matrix E. This sums up to an amount of

$$\mathcal{S}_{\mathcal{H}-Sign}(T_{I\times I}, k, n, X) \leq 2\left(2\,\mathrm{C_{sp}}\, kn(\log_2(n)+1) + 2nn_\tau(X)\right)$$

real numbers needed throughout Algorithm 6. The computational complexity is determined by the complexity of computing the formatted matrix multiplication and is therefore bounded by

$$\mathcal{N}_{\mathcal{H}-Sign}(T_{I\times I}, k, n) = \mathcal{O}\left(n \log_2^2(n) k^2\right).$$

That is, the data-sparse solver for generalized Lyapunov equations has almost linear storage requirements and a demand of flops which grows linear-polylogarithmically in n.

Algorithm 6 Calculate low-rank factor $\tilde{Y}$ of X for $AXE^T + EXA^T + BB^T = 0$.

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $E \in \mathbb{R}^{n\times n}$, tol, ϵ, τ

OUTPUT: Approximate low-rank factor Y of the solution X.

$A_0 \leftarrow (A)_{\mathcal{H}}$, $E \leftarrow (E)_{\mathcal{H}}$

$B_0 \leftarrow B$

$j = 0$

while $\|A_j + E\| > \text{tol} \cdot \|E\|$ **do**

  Compute $\mathcal{H}$-LU-factorization: $[L, U] \leftarrow LU_{\mathcal{H}}(A_j)$.

  Solve $LW = E$ by $\mathcal{H}$-forward substitution.

  Solve $VU = E$ by $\mathcal{H}$-backward substitution.

  $A_{j+1} \leftarrow \frac{1}{2}(A_j \oplus V \odot W)$

  Solve $L\tilde{B}_j = B_j$ by forward substitution.

  $B_{j+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} B_j, & V\tilde{B}_j \end{bmatrix}$

  Compute an RRLQ factorization

$$B_{j+1} = \Pi L\, Q = \Pi \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} Q$$

  with $\|L_{22}\|_2 < \tau \|B_{j+1}\|_2$ and $L_{11} \in \mathbb{R}^{r\times r}$.

  Compress columns of B_{j+1} to size r:

$$B_{j+1} \leftarrow \Pi \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}.$$

  $j = j + 1$

end while

$\tilde{Y} \leftarrow \frac{1}{\sqrt{2}} E_{\mathcal{H}}^{-1} B_{j+1}$

4.3.1 Error Bounds

We examine the influence of the introduced errors using the formatted operators in the sign function iteration on the accuracy of the computed solutions. As in the previous sections we ignore errors introduced by using an RRLQ factorization, also the scaling is not taken into account.

Notation 4.3.4 *Let j_{max} denote the maximal number of iteration steps. For the perturbed iterates $\tilde{A}_j$ arising in Algorithm 6 we write for the maximum $\mathcal{H}$-matrix inversion error (with $\tilde{E} = E_{\mathcal{H}}$)*

$$\begin{aligned} \delta &:= \max_{j=0,\ldots,j_{max}} \|\tilde{E}\tilde{A}_{\mathcal{H},j}^{-1}\tilde{E} - \tilde{E}\tilde{A}_j^{-1}\tilde{E}\|_2, \\ \delta' &:= \max_{j=0,\ldots,j_{max}} \|\tilde{E}\tilde{A}_{\mathcal{H},j}^{-1} - \tilde{E}\tilde{A}_j^{-1}\|_2, \end{aligned}$$

and for the maximum $\mathcal{H}$-matrix addition error

$$\rho := \max_{j=0,\ldots,j_{max}} \|(\tilde{A}_j \oplus \tilde{E}\tilde{A}_{\mathcal{H},j}^{-1}\tilde{E}) - (\tilde{A}_j + \tilde{E}\tilde{A}_{\mathcal{H},j}^{-1}\tilde{E})\|_2.$$

Notation 4.3.5 *For the distance between the exact and the perturbed matrices we define*

$$\begin{aligned} \epsilon_E &:= \|\tilde{E} - E\|_2, \\ \eta_0 &:= \|A_{\mathcal{H}} - A\|_2, \\ \eta_j &:= \|\tilde{A}_j - A_j\|_2, \quad \textit{for all } j = 1, \ldots, j_{max}. \end{aligned}$$

Assumption 4.3.6 *We assume with*

$$\alpha := \max_{j=0,\ldots,j_{max}} \|A_j^{-1}\|_2$$

that

$$\eta_j \, \alpha < 1, \quad \textit{for all } j = 0, \ldots, j_{max}.$$

For the perturbed iterates we obtain in Lemma 4.3.7 a bound similar to the bound (4.5) in Proposition 4.1.9 for the standard iteration with an additional factor $\|\tilde{E}\|_2^2$. The proof is similar and therefore neglected.

Lemma 4.3.7 *With Assumption 4.3.6, the forward error of the perturbed iterates in line 8 of Algorithm 6 satisfies, for all $j = 1, \ldots, j_{max}$,*

$$\eta_{j+1} = \|\tilde{A}_{j+1} - A_{j+1}\|_2 \leq \frac{1}{2} \left(\rho + \delta + \eta_j + \frac{\eta_j \alpha^2}{1 - \eta_j \alpha} \|\tilde{E}\|_2^2 \right).$$

Theorem 4.3.8 *If Assumption 4.3.6 holds, then the forward error for computing the approximate solution factors $\tilde{B}_j$ is bounded for all $j = 1, \ldots, j_{max}$ by*

$$\begin{aligned} &\|\tilde{B}_{j+1} - B_{j+1}\|_2 \\ &\quad \leq \frac{1}{\sqrt{2}} \left\{ \left(1 + \alpha \|E\|_2 + \delta' + \epsilon_E \alpha + \frac{\eta_j \alpha^2}{1 - \eta_j \alpha} \|\tilde{E}\|_2 \right) \|\tilde{B}_j - B_j\|_2 \right. \\ &\qquad \left. + \left(\delta' + \epsilon_E \alpha + \frac{\eta_j \alpha^2}{1 - \eta_j \alpha} \|\tilde{E}\|_2 \right) \|B_j\|_2 \right\}. \quad (4.14) \end{aligned}$$

Proof: Using Notation 4.3.4 and 4.3.5 and the definition of α, we obtain

$$\begin{aligned}
\|\tilde{B}_{j+1} - B_{j+1}\|_2 &\leq \frac{1}{\sqrt{2}}\Big\{\|\tilde{B}_j - B_j\|_2 + \|\tilde{E}\tilde{A}^{-1}_{\mathcal{H},j}\tilde{B}_j - EA_j^{-1}B_j\|_2\Big\} \\
&\leq \frac{1}{\sqrt{2}}\Big\{\|\tilde{B}_j - B_j\|_2 + \delta'\|\tilde{B}_j\|_2 + \|\tilde{E}\tilde{A}_j^{-1}\tilde{B}_j - \tilde{E}A_j^{-1}\tilde{B}_j\|_2 \\
&\quad + \|\tilde{E}A_j^{-1}\tilde{B}_j - EA_j^{-1}B_j\|_2\Big\} \\
&\leq \frac{1}{\sqrt{2}}\Big\{\|\tilde{B}_j - B_j\|_2 + \delta'\|\tilde{B}_j\|_2 + \|\tilde{E}\|_2\|\tilde{A}_j^{-1}(A_j - \tilde{A}_j)A_j^{-1}\|_2\|\tilde{B}_j\|_2 \\
&\quad + \|\tilde{E}A_j^{-1}\tilde{B}_j - EA_j^{-1}\tilde{B}_j\|_2 + \|EA_j^{-1}\tilde{B}_j - EA_j^{-1}B_j\|_2\Big\} \\
&\leq \frac{1}{\sqrt{2}}\Big\{\|\tilde{B}_j - B_j\|_2 + \delta'\|\tilde{B}_j\|_2 + \eta_j\alpha\|\tilde{E}\|_2\|\tilde{A}_j^{-1}\|_2\|\tilde{B}_j\|_2 \\
&\quad + \alpha\|\tilde{E} - E\|_2\|\tilde{B}_j\|_2 + \alpha\|E\|_2\|\tilde{B}_j - B_j\|_2\Big\} \\
&\overset{(4.4)}{\leq} \frac{1}{\sqrt{2}}\left\{(1 + \alpha\|E\|_2)\|\tilde{B}_j - B_j\|_2 + (\delta' + \epsilon_E\alpha + \frac{\eta_j\alpha^2}{1 - \eta_j\alpha}\|\tilde{E}\|_2)\|\tilde{B}_j\|_2\right\} \\
&\leq \frac{1}{\sqrt{2}}\left\{(1 + \alpha\|E\|_2)\|\tilde{B}_j - B_j\|_2 + (\delta' + \epsilon_E\alpha + \frac{\eta_j\alpha^2}{1 - \eta_j\alpha}\|\tilde{E}\|_2)\right. \\
&\quad \left.(\|\tilde{B}_j - B_j\|_2 + \|B_j\|_2)\right\}.
\end{aligned}$$

From this, (4.14) follows immediately. □

4.3.2 Numerical Results

The matrices A, $E \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ stem from a space discretization of the heat equation by linear finite elements as introduced in Example 4.1.13. They belong to a system in generalized state space form

$$E\dot{x}(t) = Ax(t) + Bu(t). \tag{4.15}$$

The stiffness matrix A is computed with constant diffusion coefficient $a(\cdot) \equiv 1.0$. To stop the iteration, a choice of $\text{tol} = 10^{-4}$ in the stopping criterion is shown to be useful.

The parameters ϵ and τ are chosen as in Section 4.2.2.

We observe larger resource requirements than for the algorithm in Section 4.2.2. This is caused by larger local ranks in the $\mathcal{H}$-matrix approximation of the A-iterates (and also of the derived matrices) during the iteration process. Therefore we have to restrict the problem sizes to $n = 65,536$ for Algorithm 6. The values for the computation in standard arithmetic are extrapolated for problem sizes larger than 4096. Since the complexity estimates for the formatted arithmetic are dependent on the rank distribution in the $\mathcal{H}$-matrix subblocks, see Secion 2.3, the line of the data-sparse solver in Figure 4.13

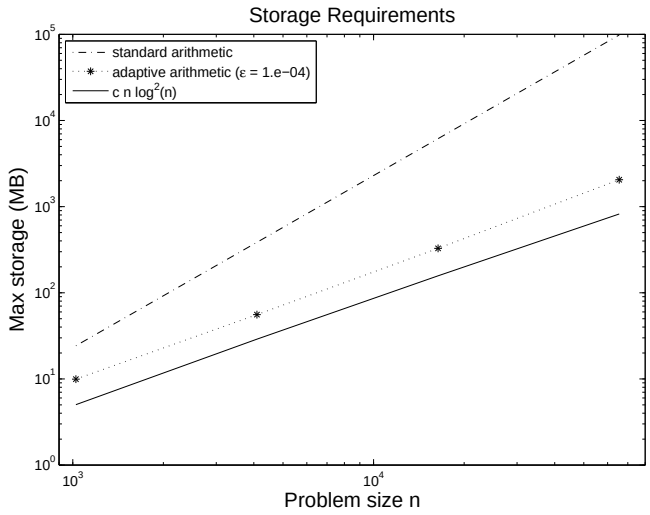


Figure 4.13: Maximum storage requirements in logarithmic scale for Algorithm 6 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line.

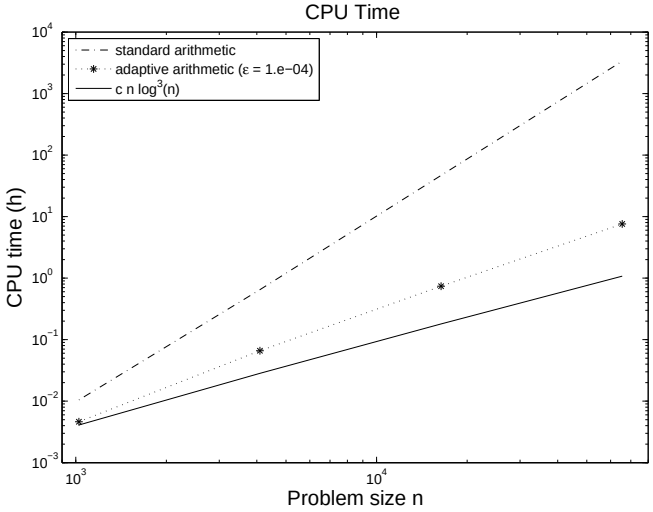


Figure 4.14: CPU time in logarithmic scale for Algorithm 6 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log_2^2(n))$ reference line.

and in Figure 4.14 is steeper than the linear-logarithmic reference line. In Table 4.9 the rank of the approximate solution factors and the accuracy of the algorithm is depicted. The results are similar to the results of Algorithm 5.

ϵ	τ	# it.	$n_\tau(X)$		time [sec]		rel. residual		rel. error
			$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
$n = 1024$									
1.e-04	1.e-04	7	12	12	16	37	1.3e-08	1.6e-09	2.0e-05
1.e-06	1.e-04	7	12	12	25	37	1.7e-09	1.6e-09	7.9e-08
1.e-08	1.e-04	7	12	12	39	37	1.7e-09	1.6e-09	2.1e-08
1.e-04	1.e-06	7	21	18	16	37	1.3e-08	1.6e-09	2.0e-05
1.e-06	1.e-06	7	18	18	25	37	1.6e-09	1.6e-09	8.0e-08
1.e-08	1.e-06	7	18	18	39	37	1.6e-09	1.6e-09	1.3e-08
1.e-04	1.e-08	7	55	25	16	37	1.3e-08	1.6e-09	2.0e-05
1.e-06	1.e-08	7	26	25	25	38	1.6e-09	1.6e-09	8.0e-08
1.e-08	1.e-08	7	25	25	39	38	1.6e-09	1.6e-09	9.5e-09
$n = 4096$									
1.e-04	1.e-04	8	14	14	237	2333	3.4e-09	4.9e-11	3.7e-05
1.e-06	1.e-04	8	14	14	530	2379	5.3e-11	4.9e-11	1.3e-07
1.e-08	1.e-04	8	14	14	1105	2358	4.9e-11	4.9e-11	1.2e-09
1.e-04	1.e-06	8	26	21	238	2354	3.4e-09	4.8e-11	3.7e-05
1.e-06	1.e-06	8	21	21	530	2324	5.2e-11	4.8e-11	1.3e-07
1.e-08	1.e-06	8	21	21	1108	2337	4.8e-11	4.8e-11	1.1e-09
1.e-04	1.e-08	8	75	29	241	2338	3.4e-09	4.8e-11	3.7e-05
1.e-06	1.e-08	8	32	29	531	2326	5.2e-11	4.8e-11	1.3e-07
1.e-08	1.e-08	8	29	29	1106	2328	4.8e-11	4.8e-11	1.1e-09
$n = 16,384$									
1.e-04	1.e-04	9	15	-	2654	-	7.6e-10	-	-
$n = 65,536$									
1.e-04	1.e-04	10	16	-	27,619	-	3.0e-10	-	-

Table 4.9: Accuracy and rank $n_\tau(X)$ of the computed solution factors from Algorithm 6 for different problem sizes and parameter combinations.

4.4 $\mathcal{H}$-Matrix Arithmetic Based Smith Iteration

In the discrete-time setting it is again the large-scale matrix $A \in \mathbb{R}^{n\times n}$ which is approximated in $\mathcal{H}$-matrix format. In the iteration scheme of the squared Smith iteration (3.38) we replace the usual matrix product in (3.38b) by a formatted multiplication

$$A_{j+1} \leftarrow A_j \odot A_j.$$

The intermediates for the solution factor are stored in usual dense format and are updated via the formatted matrix-vector product,

$$B_{j+1} \leftarrow \left[\; B_j, \quad A_j B_j \; \right].$$

The number of rows is again limited by applying an RRLQ factorization in each iteration step on B_{j+1}.

Remark 4.4.1 (**Stopping criterion**) In preliminary numerical experiments it has been observed that the convergence of the A_j's towards zero (as used in the criterion (3.35)) does not predict very well the distance between B_j and Y. Therefore, we use the stopping criterion proposed in Remark 3.3.3. We interpret

the largest singular value of B_j as converged if the relative change is bounded by the threshold for the numerical rank decision in the RRLQ factorization:

$$\frac{|\sigma_1(B_j) - \sigma_1(B_{j+1})|}{\sigma_1(B_j)} < \tau.$$

The numerical experiments support the use of this choice. □

All computational steps of the data-sparse solver are summarized in Algorithm 7.

Algorithm 7 Calculate low-rank factor $\tilde{Y}$ of X for $AXA^T - X + BB^T = 0$.

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, ϵ, τ
OUTPUT: Approximate low-rank factor S of the solution X.

1: $A_0 \leftarrow (A)_{\mathcal{H}}$
2: $B_0 \leftarrow B$
3: $j = 0$
4: **while** $|\sigma_1(B_j) - \sigma_1(B_{j+1})|/\sigma_1(B_j) \geq \tau$ **do**
5: $B_{j+1} \leftarrow \left[\begin{array}{cc} B_j, & A_jB_j \end{array}\right]$
6: Compute an RRLQ factorization

$$B_{j+1} = \Pi L\, Q = \Pi \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} Q$$

with $\|L_{22}\|_2 < \tau\|B_{j+1}\|_2$ and $L_{11} \in \mathbb{R}^{r\times r}$.
7: Compress columns of B_{j+1} to size r:

$$B_{j+1} \leftarrow \Pi \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}.$$

8: $A_{j+1} \leftarrow A_j \odot A_j$
9: $j = j + 1$
10: **end while**
11: $\tilde{Y} \leftarrow B_{j+1}$

Complexity. With the usual technical assumptions on the underlying $\mathcal{H}$-tree, the required workspace for the data-sparse Smith iteration is given by the sum of the workspace for the $\mathcal{H}$-matrix (and iterates) A_j and the solution factor B_j. With the bound for A as stated in (4.1) the amount on storage is bounded by

$$\mathcal{S}_{\mathcal{H}-\mathrm{Smith}}(T_{I\times I}, k, n, X) \leq 2(\mathrm{C}_{\mathrm{sp}}\, kn(\log_2(n) + 1) + 2nn_\tau(X)).$$

The complexity of each iteration in (3.38a) is bounded by the complexity of the formatted multiplication and is thus of linear-polylogarithmic dependency on n:

$$\mathcal{N}_{\mathcal{H}-\mathrm{Smith}}(T_{I\times I}, k, n) = \mathcal{O}(n\log_2^2(n)k^2).$$

4.4.1 Error Bounds

Notation 4.4.2 *The maximum error for the formatted multiplication is denoted by*

$$\rho := \max_{j=0,\ldots,j_{max}} \|\tilde{A}_j \odot \tilde{A}_j - \tilde{A}_j\, \tilde{A}_j\|_2.$$

Notation 4.4.3 *For the distance between the exact and the perturbed matrices we define*

$$\begin{aligned} \eta_0 &:= \|A_{\mathcal{H}} - A\|_2, \\ \eta_j &:= \|\tilde{A}_j - A_j\|_2, \quad \textit{for all } j = 0, \ldots, j_{max}. \end{aligned}$$

Theorem 4.4.4 *The forward error for computing the approximate solution factors $\tilde{B}_j$ by Algorithm 7 is bounded for all $j = 1, \ldots, j_{max}$ by*

$$\begin{aligned} \|\tilde{B}_{j+1} - B_{j+1}\|_2 &\leq (1 + \|\tilde{A}_j\|_2)\|\tilde{B}_j - B_j\|_2 + \eta_j \|B_j\|_2 \\ &\leq \sum_{\ell=0}^{j} \left(\prod_{i=\ell+1}^{j} (1 + \|\tilde{A}_i\|_2) \right) \eta_\ell \, \|B_\ell\|_2. \end{aligned} \tag{4.16}$$

For the relative errors we have

$$\frac{\|\tilde{B}_{j+1} - B_{j+1}\|_2}{\|B_{j+1}\|_2} \leq \sum_{\ell=0}^{j} \eta_\ell \prod_{i=\ell+1}^{j} (1 + \|\tilde{A}_i\|_2).$$

Proof: Using Notation 4.4.2 and 4.4.3, we obtain

$$\begin{aligned} \|\tilde{B}_{j+1} &- B_{j+1}\|_2 \\ &\leq \|\tilde{B}_j - B_j\|_2 + \|\tilde{A}_j \tilde{B}_j - A_j B_j\|_2 \\ &\leq \|\tilde{B}_j - B_j\|_2 + \|\tilde{A}_j\|_2 \|\tilde{B}_j - B_j\|_2 + \|\tilde{A}_j - A_j\|_2 \|B_j\|_2 \\ &\leq (1 + \|\tilde{A}_j\|_2)\|\tilde{B}_j - B_j\|_2 + \eta_j \|B_j\|_2. \end{aligned}$$

The second inequality of the statement (4.16) is now immediate. For the bound of the relative error, the same arguments as in the proof of Corollary 4.2.3 hold. □

4.4.2 Numerical Results

We consider a time discretization of the instationary heat equation (4.13) ($a(\cdot) \equiv 1.0$) with time step size $T_s = 10^{-4}$. Using the FEM space discretization as introduced in Example 4.1.13 and a backward Euler scheme for the time discretization, we obtain a discrete, time-invariant system

$$\underbrace{(E + T_s \hat{A})}_{\tilde{A}} \hat{x}_{k+1} = E\hat{x}_k + T_s \hat{B} u_k, \quad k \in \mathbb{N}.$$

We use the formatted $\mathcal{H}$-LU decomposition to invert the matrix $\tilde{A}$ and obtain a discrete-time system in standard form,

$$x_{k+1} = \underbrace{(E + T_s\hat{A})^{-1}_{\mathcal{H}} E}_{A} x_k + \underbrace{T_s(E + T_s\hat{A})^{-1}_{\mathcal{H}} \hat{B}}_{B} u_k, \quad k \in \mathbb{N}.$$

In our example, the resulting state matrix $A \in \mathbb{R}^{n\times n}$ is stable and $B \in \mathbb{R}^n$. The results of the data-sparse solver using different values for the blockwise accuracy ϵ is compared with a standard dense implementation. Due to limited workspace, the squared Smith iteration using the dense matrix format can only be applied to problems up to size $n = 4096$. To compare the accuracy of these calculations, the relative residual

$$R(X) = \frac{\|AXA^T - X + BB^T\|}{\|A\|^2\|X\| + \|X\| + \|B\|^2}$$

is computed by the Frobenius norm and two "economy-size" QR factorizations

$$\begin{aligned} \|A\tilde{Y}\tilde{Y}^T A^T - \tilde{Y}\tilde{Y}^T + BB^T\|_F &= \| \left[A\tilde{Y}, -\tilde{Y}, B \right] \left[A\tilde{Y}, \tilde{Y}, B \right]^T \|_F \\ &= \|R_1 R_2^T\|_F. \end{aligned}$$

For the numerical rank decision we vary the parameter τ from 10^{-4} to 10^{-8}. The influence of τ on the accuracy as well as on the computational complexity is seen in Table 4.10. Again it is observed that the accuracy of the solution computed by formatted arithmetic is mainly determined by the parameter ϵ. A decrease of τ results in larger ranks for the solution without improving the accuracy of the solution of the data-sparse approach. The relative residual of the standard dense solver gets smaller for decreasing τ. Note that τ is also used in the stopping criterion. It is seen that the complexity is significantly smaller for the data-sparse solver than for the standard squared Smith iteration.

ϵ	τ	# it.	$n_\tau(X)$		time[sec]		rel. residual		rel. error
			$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
$n = 1024$									
1.e-04	1.e-04	11	11	11	3	7	7.8e-09	8.5e-11	1.3e-05
1.e-06	1.e-04	11	11	11	6	7	1.6e-10	8.5e-11	4.3e-07
1.e-08	1.e-04	11	11	11	7	11	8.5e-11	8.5e-11	1.1e-09
1.e-04	1.e-06	11	19	17	3	7	7.8e-09	1.1e-13	1.3e-05
1.e-06	1.e-06	11	17	17	6	8	1.4e-10	1.1e-13	4.3e-07
1.e-08	1.e-06	11	17	17	11	8	7.5e-13	1.1e-13	1.1e-09
1.e-04	1.e-08	12	40	23	3	9	7.8e-09	2.7e-17	1.3e-05
1.e-06	1.e-08	12	25	23	6	9	1.4e-10	2.7e-17	4.3e-07
1.e-08	1.e-08	12	23	23	11	9	7.4e-13	2.7e-17	1.1e-09
$n = 4096$									
1.e-04	1.e-04	11	12	12	69	431	1.5e-08	3.1e-11	2.6e-05
1.e-06	1.e-04	11	12	12	141	432	1.2e-10	3.11e-11	3.3e-07
1.e-08	1.e-04	11	12	12	259	432	3.11e-11	3.11e-11	1.13e-09
1.e-04	1.e-06	11	20	18	69	437	1.46e-08	7.6e-14	2.6e-05
1.e-06	1.e-06	11	18	18	141	430	1.2e-10	7.6e-14	3.3e-07
1.e-08	1.e-06	11	18	18	260	436	1.3e-12	7.6e-14	1.1e-09
1.e-04	1.e-08	12	46	23	74	478	1.5e-08	1.3e-17	2.6e-05
1.e-06	1.e-08	12	24	23	146	470	1.2e-10	1.3e-17	3.3e-07
1.e-08	1.e-08	12	23	23	270	466	1.3e-12	1.3e-17	1.1e-09
$n = 16,384$									
1.e-04	1.e-04	10	12	-	637	-	1.7e-08	-	-
$n = 65,536$									
1.e-04	1.e-04	11	11	-	4924	-	1.7e-08	-	-
$n = 262,144$									
1.e-04	1.e-04	11	11	-	35,539	-	1.3e-08	-	-

Table 4.10: Accuracy and rank $n_\tau(X)$ of the computed solution factors of Algorithm 7 for different problem sizes and parameter combinations.

Chapter 5

Model Reduction for Data-Sparse Systems

With the ever-developing technologies in various engineering applications (microelectronics, micro-electro-mechanical systems (MEMS), electromagnetism, fluid dynamics, control design, etc.), more and more mathematical systems with very large dimensions have to be simulated and solved. Traditional numerical simulation methods are powerless in solving systems of very large dimensions due to limited computer memory and CPU consumption. Model order reduction has been proved to be very promising to enhance the efficiency of traditional numerical simulation methods. It is also very important in control and optimization problems for partial differential equations. There, the associated large-scale systems have to be solved very often. To solve these problems in reasonable time it is absolutely necessary to reduce the dimension of the underlying system. Furthermore, in modern (LQG-, $\mathcal{H}_2$-, $\mathcal{H}_\infty$-) feedback control, the order of the controller is typically comparable with the dimension of the underlying parabolic control system. Thus, for practical use, low order controllers should be constructed from reduced-order systems. For these problems the system matrices are typically large-scale (often $n \geq \mathcal{O}(10^5)$) and sparse as they stem from the spatial discretization of some partial differential operator. On the other hand, boundary element discretizations of integral equations lead to large-scale *dense* matrices that often have a data-sparse representation [89, 147, 148]. Usually, the number of inputs and outputs in practical applications is small compared to the number of states, so that it is reasonable to assume $m, p \ll n$ from now on.

Model reduction aims at eliminating some of the state variables of the original large-scale system. Considering again the LTI system (2.1), the task in model reduction is to find another LTI system

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{B}u(t), \qquad \hat{x}(0) = \hat{x}_0, \tag{5.1a}$$

$$\hat{y}(t) = \hat{C}\hat{x}(t) + \hat{D}u(t), \tag{5.1b}$$

with reduced state space dimension $r \ll n$ and $\hat{A} \in \mathbb{R}^{r\times r}$, $\hat{B} \in \mathbb{R}^{r\times m}$, $\hat{C} \in \mathbb{R}^{p\times r}$, $\hat{D} \in \mathbb{R}^{p\times m}$ and initial value $\hat{x}_0 \in \mathbb{R}^r$. The LTI system of order r is denoted by

$\Sigma(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ in the following. In the discrete-time setting we are looking for a reduced-order system

$$\hat{x}_{k+1} = \hat{A}\hat{x}_k + \hat{B}u_k, \qquad \hat{x}_0 = \hat{x}^0, \tag{5.2a}$$

$$\hat{y}_k = \hat{C}\hat{x}_k + \hat{D}u_k, \tag{5.2b}$$

of the same reduced complexity r. The dimensions of the matrices are equal to those in the continuous-time setting.

Projection-based model order reduction techniques for LTI systems can be divided roughly into two main classes:

1. Methods based on Krylov subspaces;
2. SVD-based methods.

Both classes are based on state space projection in the following sense. We consider a change of basis in state space $\tilde{x} = Tx$, $T \in \mathbb{R}^{n\times n}$ nonsingular. Partitioning T and $\tilde{x}$ as

$$T = \begin{bmatrix} T_l \\ * \end{bmatrix}, \quad T^{-1} = [\, T_r,\ * \,], \quad \tilde{x} = \begin{bmatrix} \hat{x} \\ * \end{bmatrix}, \quad \text{with } T_l^T, T_r \in \mathbb{R}^{n\times r},\ \hat{x} \in \mathbb{R}^r,$$

$T_r T_l \in \mathbb{R}^{n\times n}$ is an oblique projector onto the r dimensional subspace spanned by the columns of T_r along the kernel of T_l. Note that T_l and T_r are biorthogonal, that is,

$$T_l T_r = I_r.$$

The other part of the transformed state $\tilde{x}$ denoted by $*$ is neglected, the state is approximated by $\hat{x} = T_l x$. A reduced-order system is constructed by a Petrov-Galerkin projection

$$T_l(T_r\dot{\hat{x}}(t) - AT_r\hat{x} - Bu(t)) = 0,$$

resulting in the following matrices in (5.1) and in (5.2)

$$\hat{A} := T_l A T_r,\ \hat{B} := T_l B,\ \hat{C} := C T_r. \tag{5.3}$$

The first important class of model order reduction methods for linear systems is based on Krylov subspaces, basically implemented as moment matching methods. These methods are very efficient in many engineering applications, originally they were derived for the use in circuit simulation. The main idea is to approximate the rational TFM $G(\cdot)$ by a rational function of lower degree. To do this, $G(\cdot)$ is expanded into a series around one or more expansion points of the complex plane or at infinity. The approximate transfer function $\hat{G}(\cdot)$ is computed such that it matches some of the coefficients of the expansion series. For systems in state space form (2.1), the problem reduces to the task of computing orthonormal bases for the Krylov subspaces $\mathcal{C}_r(A, B)$ and $\mathcal{O}_r(A, C)$ of order $1 \le r \le n$ (see the definitions in (2.11) and (2.14), respectively). It is well known that this can be done in a numerically efficient way by the iterative Arnoldi or Lanczos methods. After r steps of iteration the methods produce

either one transformation matrix with r orthonormal columns for a Galerkin projection, or the biorthogonal transformation matrices $T_l^T, T_r \in \mathbb{R}^{n\times r}$ for applying a Petrov-Galerkin method to (2.1). In these iterative algorithms only matrix-vector multiplications are used which are simple to implement and the complexity of the resulting methods is only $\mathcal{O}(n^2r)$ for dense, $\mathcal{O}(nr^2)$ for sparse systems. To overcome the numerical ill-conditioning of the early approaches as the asymptotic waveform evaluation [140], some recent research leads to numerically more robust methods as Padé via Lanczos [62] (see also [63, 67]). For matching the transfer function over a large frequency range, rational interpolation methods including multiple interpolation points as the dual rational Arnoldi and the rational Lanczos method were proposed [81]. In general, these methods do not preserve important properties of the original system as stability and passivity. For some reduced systems this can be avoided applying post-processing techniques [10, 11]. Using restarting techniques as described in [82] or the implicitly restarted dual Arnoldi method [103], a subsystem is retained as reduced-order model which has purely stable eigenvalues. Earlier results concerning error bounds for Krylov subspace based methods [12] yield only local bounds for the transfer function. Such bounds can only be used to estimate the accuracy of the transfer function in a certain frequency range, it cannot give a global estimation in the whole frequency domain. In a new work of Gugercin, Antoulas and Beattie [84] it is shown that for stable systems the $\mathcal{H}_2$-error is minimized for SISO systems if the expansion points are chosen as mirror images of the reduced-order system poles. The numerical results in this paper also indicate small $\mathcal{H}_\infty$-errors but there is no theoretical analysis for the existence of an $\mathcal{H}_\infty$-error bound. For survey papers on model reduction based on Krylov subspaces see [7, 66].

In systems theory and control of ordinary or partial differential equations, balanced truncation [127, 129] and related methods are the methods of choice since they have some desirable properties: they preserve the stability of the system [139] and provide a global computable error bound [60, 72] which allows an adaptive choice of the reduced order r. The basic approach relies on balancing the two system Gramians of (2.1) or of (2.23), respectively. A variant of the classical balanced truncation method, called cross-Gramian approach, is based on the solution of one Sylvester equation. Thus, the major part of the computational complexity of these methods stems from the solution of these large-scale matrix equations. As described in Chapter 3, over the last few years the computational efficiency of solvers for linear matrix equations was considerably improved by computing low-rank approximations to the system Gramians. Most of the methods are shown to be applicable to medium-to-large-scale systems (with n in the range from $\mathcal{O}(10^3)$ to $\mathcal{O}(10^4)$) if the system matrix A is sparse. As some of the iterative methods allow for parallel computing, larger sparse problems with up to $\mathcal{O}(10^5)$ states can be solved on computers with distributed memory architecture [6, 31]. The use of efficient solvers for large-scale Lyapunov equations improves the implementations of balanced truncation, e.g. [29, 32, 36, 37, 85, 143, 163], and of balancing-related model reduction methods as optimal Hankel norm approximation [72], frequency weighted balanced truncation [60, 71, 83] and balanced stochastic truncation [34]. For a survey

on balanced truncation model reduction see [83]. For generalized state space systems, where the matrix E might be singular, there exists generalizations of balanced truncation, see for instance [6, 30, 121, 126, 138, 159, 160]. If a good approximation of the steady-state is required singular perturbation approximation is proposed [33, 122, 165]. SPA computes a reduced-order system with good matching of the TFM at low frequencies (which corresponds to small steady-state errors) on the basis of a minimal and balanced realization of the original system. Model reduction based on the cross-Gramian was considered in several works [1, 3, 68, 156]. In [1] the reduced-order system is computed from the eigenspaces associated with large eigenvalues of the $n \times n$ Gramian. This is computationally very demanding and thus fails for the problems considered in this work. The approaches in [3, 68, 156] belong to the class of Krylov projection methods as they iteratively compute low-rank approximations to the cross-Gramian by an implicitly restarted Arnoldi method. An approximately balanced reduced-order system is obtained by a partial eigenvalue decomposition of this Gramian. The resulting algorithm is quite expensive and technically very involved if applied to MIMO systems.

All SVD-based methods do not exploit the structural information of the underlying problem as the description of a physical system by a set of ODEs or PDEs. Furthermore, all the methods above fail for the application to large-scale, dense LTI systems. We overcome these limitations for the practically relevant class of large-scale, dense systems which are assumed to have a data-sparse representation. We incorporate the data-sparse solvers for matrix equations derived in the last chapter in the algorithms for model order reduction. This is done in Section 5.1 for the usual balanced truncation implementation, in Section 5.2 for the singular perturbation approximation and in Section 5.3 for a cross-Gramian approach. Note that preliminary results are published in [18].

5.1 Model Reduction by Balanced Truncation

The first time that the concept of balancing was proposed in the literature was in 1976 by Mullis and Roberts [129]. Some years later Moore [127] recognized and persued the important system theoretic background of the method. The main principle of balanced truncation and of balancing-related model reduction is finding a particular state space basis in which we can easily determine the states, which will be truncated. These states should have small impact on the system behavior concerning both, reachability and observability. Such a system representation, where states which are difficult to observe are also difficult to reach and vice-versa, is obtained by a balancing transformation. The required state space transformation, $x \to Tx$, $T \in \mathbb{R}^{n \times n}$ nonsingular, leads to a balanced realization of the original system

$$(A, B, C, D) \leftarrow (TAT^{-1}, TB, CT^{-1}, D),$$

where the reachability Gramian $\mathcal{P}$ and the observability Gramian $\mathcal{Q}$ are equal and diagonal:

$$\mathcal{P} = \mathcal{Q} = \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix}, \quad \sigma_1 \geq \cdots \geq \sigma_n > 0.$$

For minimal systems there always exist balancing transformations and by the positive definiteness of $\mathcal{P}$ and $\mathcal{Q}$, all eigenvalues of $\mathcal{PQ}$ are strictly positive numbers. The square roots of these eigenvalues are known to be the HSVs of the system (2.1):

$$\Lambda(\mathcal{PQ}) = \{\sigma_1^2, \ldots, \sigma_n^2\}.$$

Note that balanced truncation is also applicable to non-minimal, stable systems. There, the Gramians are positive semi-definite and some of the HSVs are of zero magnitude, see [161]. To simplify the theory, we restrict to minimal systems in the following. The HSVs provide a systematic way to identify the states which are least involved in the energy transfer from past inputs to future outputs. For a system in balanced coordinates the energy interpretation in (2.17) can simply be computed by

$$E = \frac{x_0^T \mathcal{P}^{1/2} \mathcal{Q} \mathcal{P}^{1/2} x_0}{x_0^T x_0} = \frac{1}{\|x_0\|^2} \sum_{j=1}^{n} \sigma_j^2 (x_0)_j^2.$$

Thus, the states which have a small impact on the energy transfer are determined as those which correspond to small HSVs. If the states corresponding to the $n-r$ smallest HSVs were truncated from a balanced realization, a reduced-order model (5.1) of size r is obtained. An a priori computable global error bound for the $\mathcal{H}_\infty$-norm error between the TFM $G(\cdot)$ of the original system and the TFM associated to (5.1)

$$\hat{G}(s) = \hat{C}(sI_r - \hat{A})^{-1}\hat{B} + \hat{D}$$

exists [60, 72]:

$$\|G - \hat{G}\|_{\mathcal{H}_\infty(\mathbb{C}^+ \to \mathbb{C}^{p \times m})} \leq 2 \sum_{j=r+1}^{n} \sigma_j. \tag{5.4}$$

Using the estimates (2.21) and (2.22), the worst output error is bounded in time domain

$$\|y - \hat{y}\|_{\mathcal{L}_2([0,\infty) \to \mathbb{R}^p)} \leq 2 \left(\sum_{j=r+1}^{n} \sigma_j \right) \|u\|_{\mathcal{L}_2([0,\infty) \to \mathbb{R}^m)} \tag{5.5}$$

and in frequency domain

$$\|Y - \hat{Y}\|_{\mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^p)} \leq 2 \left(\sum_{j=r+1}^{n} \sigma_j \right) \|U\|_{\mathcal{H}_2(\mathbb{C}^+ \to \mathbb{C}^m)}. \tag{5.6}$$

Thus, the error bound (5.4) provides a nice way to adapt the selection of the reduced order corresponding to a prespecified error tolerance:

$$\min \left\{ r \in \mathbb{N} \,|\, 2 \sum_{j=r+1}^{n} \sigma_j \leq \text{tol} \right\}. \tag{5.7}$$

In addition, the reduced-order system remains stable [139] and balanced with HSVs σ_1 to σ_r of the original system.

There are several approaches for computing the balancing transformation matrix T as well as the truncation matrices T_l and T_r for model order reduction following (5.3). A numerically robust and efficient method is the *square root method* (SR method) which is used in the next section. The SR method of balanced truncation is based on the Cholesky factors S, $R \in \mathbb{R}^{n\times n}$ of the Gramians $\mathcal{P} = SS^T$ and $\mathcal{Q} = RR^T$. It can also be applied to non-minimal systems where we have $\text{rank}\,(S) < n$ and/or $\text{rank}\,(R) < n$, see [115, 161].

A similarity relation between the product of the Cholesky factors and the square root of the Gramian product, $(\mathcal{PQ})^{1/2} \sim S^T R$, suggests to compute an SVD of $S^T R$,

$$S^T R = U\Sigma V^T, \quad U,\, V \in \mathbb{R}^{n\times n},\ \Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n), \tag{5.8}$$

to obtain a balancing transformation. To see that the matrix

$$T = \Sigma^{-1/2} V^T R^T \tag{5.9}$$

transforms the system into balanced coordinates, note that the inverse of T is given by

$$T^{-1} = SU\Sigma^{-1/2}$$

as

$$\Sigma^{-1/2} V^T R^T SU \Sigma^{-1/2} = \Sigma^{-1/2} V^T V \Sigma U^T U \Sigma^{-1/2} = I_n,$$

and thus

$$\begin{aligned} T\mathcal{P}T^T &= \Sigma^{-1/2} V^T R^T SS^T RV \Sigma^{-1/2} \\ &= \Sigma^{-1/2} V^T V \Sigma U^T U \Sigma V^T V \Sigma^{-1/2} &= \Sigma, \\ T^{-T}\mathcal{Q}T^{-1} &= \Sigma^{-1/2} U^T S^T RR^T SU \Sigma^{-1/2} \\ &= \Sigma^{-1/2} U^T U \Sigma V^T V \Sigma U^T U \Sigma^{-1/2} &= \Sigma. \end{aligned}$$

It is also observed [115, 161] that not the complete transformation matrix $T \in \mathbb{R}^{n\times n}$ needs to be computed. To show this, the SVD is partitioned as follows:

$$S^T R = \begin{bmatrix} U_1, & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}, \text{ with } \Sigma_1 = \text{diag}(\sigma_1, \ldots, \sigma_r). \tag{5.10}$$

We assume that the singular values $\{\sigma_1, \ldots, \sigma_n\}$, which equal the HSVs of the system, are in descending order. If there is a significant gap between σ_r

and σ_{r+1}, $\sigma_r \gg \sigma_{r+1}$, the splitting in (5.10) seems natural. The balancing transformation (5.9) is computed corresponding to (5.10) by

$$T = \begin{bmatrix} \Sigma_1^{-1/2} V_1^T \\ \Sigma_2^{-1/2} V_2^T \end{bmatrix} R^T, \quad T^{-1} = S \, [\, U_1 \Sigma_1^{-1/2}, \quad U_2 \Sigma_2^{-1/2} \,].$$

Applying only the parts

$$T_l = \Sigma_1^{-1/2} V_1^T R^T \in \mathbb{R}^{r \times n}, \qquad T_r = S U_1 \Sigma_1^{-1/2} \in \mathbb{R}^{n \times r} \tag{5.11}$$

of the balancing transformation T, the system is transformed into balanced coordinates and, simultaneously, the states corresponding to small HSVs are truncated. Note that $T_l T_r = I_r$ and $T_r T_l$ defines an oblique projection. The resulting reduced-order system

$$(\hat{A}, \hat{B}, \hat{C}, \hat{D}) = (T_l A T_r, \, T_l B, \, C T_r, \, D)$$

is stable and of order r and, additionally, the worst output error is bounded in time (5.5) and in frequency domain (5.6).

An efficient implementation of this method is proposed in [32] where the solution factors are computed as full-rank factors:

$$S \in \mathbb{R}^{n \times \operatorname{rank}(\mathcal{P})}, \quad R \in \mathbb{R}^{n \times \operatorname{rank}(\mathcal{Q})}.$$

This is of particular interest in large-scale computation if the Gramians have low rank at least numerically: $n_\tau(\mathcal{P})$, $n_\tau(\mathcal{Q}) \ll n$. Then it is sufficient to compute an economy-size SVD in (5.8) with $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_{n_\tau(\mathcal{Q})})$, $U \in \mathbb{R}^{n_\tau(\mathcal{P}) \times n_\tau(\mathcal{Q})}$ and $V \in \mathbb{R}^{n_\tau(\mathcal{Q}) \times n_\tau(\mathcal{Q})}$ for the case of $n_\tau(\mathcal{P}) \geq n_\tau(\mathcal{Q})$. The case $n_\tau(\mathcal{P}) < n_\tau(\mathcal{Q})$ can be treated analogously. An additional benefit of this ansatz is that all computational costs are of reduced complexity during the computation of the reduced-order system as soon as the matrix equations

$$A\mathcal{P} + \mathcal{P}A^T + BB^T = 0, \qquad A^T\mathcal{Q} + \mathcal{Q}A + C^T C = 0 \tag{5.12}$$

are solved. We list the complexity of the computational steps at the end of Section 5.1.1. Balanced truncation methods for discrete-time LTI systems (2.23) are performed analogously to the continuous-time case. The only difference is the computation of the two Gramians which are in the discrete-time setting the solutions of two Stein equations

$$A\mathcal{P}A^T - \mathcal{P} + BB^T = 0, \qquad A^T\mathcal{Q}A - \mathcal{Q} + C^T C = 0. \tag{5.13}$$

Note that in the discrete-time case the reduced-order models are in general not balanced [131, Section 1.9].

5.1.1 Approximate Balanced Truncation

To make balanced truncation applicable to a wider class of large-scale problems, we use the $\mathcal{H}$-matrix based Lyapunov solver as described in Algorithm 5 in the SR method for balanced truncation. Thus, the computational cost in the most

demanding part of balanced truncation is reduced to linear-logarithmic complexity as derived in the last chapter. Solving both Lyapunov equations (5.12) simultaneously reduces the computational complexity as well as the storage further since both iteration schemes contain the same A-iterates. For discrete-time systems the original solver for the Stein equations (5.13) is replaced by the $\mathcal{H}$-matrix based Smith iteration as described in Algorithm 7. Again it is advised to couple the computation of both equations since one A-iteration is redundant. The data-sparse solvers compute approximate full-rank factors of the Gramians:

$$\mathcal{P} \approx \tilde{S}\tilde{S}^T, \quad \mathcal{Q} \approx \tilde{R}\tilde{R}^T, \qquad \text{with } \tilde{S} \in \mathbb{R}^{n \times n_\tau(\mathcal{P})},\ \tilde{R} \in \mathbb{R}^{n \times n_\tau(\mathcal{Q})}.$$

Since computing an SVD is not available in formatted arithmetic, the solution factors $\tilde{S}$ and $\tilde{R}$ are computed in dense format. In the following, the minimum of $n_\tau(\mathcal{P})$ and $n_\tau(\mathcal{Q})$ is denoted by $\tilde{n}$. Note that using $\tilde{S}$ and $\tilde{R}$ in (5.10) reduces the computable part of the original BT error bound (5.4) to

$$\delta := 2 \sum_{j=r+1}^{\tilde{n}} \sigma_j \tag{5.14}$$

since only the first $\tilde{n}$ HSVs are computed in an economy-size SVD. Thus, the estimate (5.4) is under-estimated if $\tilde{n} < n$. Usually, $\tilde{n}$ equals the numerical rank of $S^T R$ and can thus be considered as a "numerical McMillan degree".

Moreover, an error analysis in [85, 86] suggests that the error in the computed bound δ, introduced by using approximate low-rank factors, is also affected by the square of the condition number of T, where $A = T\Lambda T^{-1}$ is a spectral decomposition of A. Hence, for ill-conditioned T, the computed error bound may under-estimate the model reduction error significantly.

As discussed in Chapter 4, we can take influence on the accuracy of the computed low-rank factors $\tilde{S}$ and $\tilde{R}$ by the choice of the two parameters τ and ϵ in the data-sparse solvers. As already noted, it is sufficient to choose $\tau \leq \sqrt{\epsilon}$ to obtain a relative error in the approximate Gramians in the size of ϵ. But for the purpose of balanced truncation, we need $\tau \sim \epsilon$ as the accuracy of the reduced-order model is affected by the accuracy of the solution factors themselves: we may assume that Algorithms 5 and 7 yield $\tilde{S}$ and $\tilde{R}$, so that $S = \left[\ \tilde{S},\ \ E_S\ \right]$ and $R = \left[\ \tilde{R},\ \ E_R\ \right]$, where $\|E_S\|_2 \leq \tau\|S\|_2$, $\|E_R\|_2 \leq \tau\|R\|_2$. Then

$$S^T R = \begin{bmatrix} \tilde{S}^T\tilde{R} & \tilde{S}^T E_R \\ E_S^T\tilde{R} & E_S^T E_R \end{bmatrix}.$$

Hence, the relative error introduced by using the "small" SVD, i.e., that of $\tilde{S}^T\tilde{R}$, rather than the full SVD, i.e., that of $S^T R$, is proportional to τ. Therefore, a choice of $\tau = \sqrt{\epsilon}$ would lead to an error of size $\sqrt{\epsilon}$ in the computed Hankel singular values as well as in the projection matrices T_l, T_r and thus in the reduced-order model.

Remark 5.1.1 (Stability) In general, using low-rank approximations to the system Gramians it can not be assured that model reduction by balanced truncation still preserves the stability of the original system [85, Section 2.6]. However, in practice, we have never encountered this phenomenon and in many

Algorithm 8 Approximate Balanced Truncation for LTI systems (2.1), (2.23)

INPUT: $A_{\mathcal{H}} \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, tol, τ, ϵ.
OUTPUT: $\hat{A} \in \mathbb{R}^{r\times r}$, $\hat{B} \in \mathbb{R}^{r\times m}$, $\hat{C} \in \mathbb{R}^{p\times r}$, $\hat{D} \in \mathbb{R}^{p\times m}$; reduced order r, error bound δ.

1: Compute low-rank factors $\tilde{S} \in \mathbb{R}^{n\times n_\tau(\mathcal{P})}$, $\tilde{R} \in \mathbb{R}^{n\times n_\tau(\mathcal{Q})}$ of the system Gramians using Algorithm 5 for continuous-time systems, Algorithm 7 in the discrete-time case.

2: Compute SVD of $\tilde{S}^T\tilde{R}$ ($\tilde{n} := \min\{n_\tau(\mathcal{P}),\, n_\tau(\mathcal{Q})\}$)

$$\tilde{S}^T\tilde{R} = [\,U_1,\; U_2\,] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix},$$

with $\Sigma_1 = \operatorname{diag}(\sigma_1, \ldots, \sigma_r)$, $\Sigma_2 = \operatorname{diag}(\sigma_{r+1}, \ldots, \sigma_{\tilde{n}})$, $\sigma_r > \sigma_{r+1}$. Adaptive choice of r by tol: $\delta = 2 \sum_{j=r+1}^{\tilde{n}} \sigma_j \leq \text{tol}$.

3: Compute truncation matrices: $T_l = \Sigma_1^{-\frac{1}{2}} V_1^T \tilde{R}^T,\; T_r = \tilde{S}\, U_1\, \Sigma_1^{-\frac{1}{2}}$.

4: Compute BT reduced-order model:

$$\hat{A} = T_l A_{\mathcal{H}} T_r,\; \hat{B} = T_l B,\; \hat{C} = C T_r,\; \hat{D} = D.$$

other papers where approximate Gramians were used for model order reduction, see for instance [86, 118, 135], instability is never observed. Moreover, for really large-scale problems a detailed knowledge about the spectrum is rarely available in advance. But, if the order of the system is significantly reduced by approximate balanced truncation it is simple to compute the eigenvalues of $\hat{A}$ and thus checking stability. For the case that unstable eigenvalues exist it is possible to split the unstable part of the reduced-order system by computing one RRQR decomposition of $I_r - \operatorname{sign}(\hat{A})$ and solving one Sylvester equation to block-diagonalize $\hat{A}$, see for instance [29, Algorithm 2]. □

Complexity. As derived in Section 4.2, using the formatted arithmetic in Algorithm 5 (respectively in Algorithm 7) reduces the computational complexity in the first stage of Algorithm 8 from $\mathcal{O}(n^3)$ to $\mathcal{O}(n \log_2^3(n) k^2)$. The iterations need $\mathcal{O}(n \log_2(n) k + n(n_\tau(\mathcal{P}) + n_\tau(\mathcal{Q})))$ storage. Computing the SVD of $\tilde{S}^T\tilde{R}$ requires $2n_\tau(\mathcal{P})n_\tau(\mathcal{Q})n$ flops for the matrix-matrix multiplication and $14n_\tau(\mathcal{P})n_\tau(\mathcal{Q})^2 + 8n_\tau(\mathcal{Q})^3$ flops, assuming $n_\tau(\mathcal{P}) \geq n_\tau(\mathcal{Q})$, for the Golub-Reinsch SVD [74, Section 5.4.5]. The workspace requirements for this step are $n_\tau(\mathcal{P})^2 + n_\tau(\mathcal{Q})^2$ real numbers. The truncation matrices T_l and T_r are obtained by computing matrix products of parts of the SVD. This can be done by $2rn(n_\tau(\mathcal{P}) + n_\tau(\mathcal{Q}))$ flops. Additional workspace of size $2rn$ is needed. The reduced-order system (5.1) (respectively (5.2)) is derived by applying a Petrov-Galerkin projection to the original system $\Sigma(A, B, C, D)$. Note, that the projection is applied to $A_{\mathcal{H}}$ and thus, the costs for computing $T_l A_{\mathcal{H}} T_r$ are reduced to $\mathcal{O}(rn \log_2(n) k)$. $\hat{B}$ and $\hat{C}$ are computed in $2rnm + 2pnr$ operations.

To summarize, if the numerical ranks $n_\tau(\mathcal{P})$, $n_\tau(\mathcal{Q})$ of the Gramians are small compared to n, none of the steps besides solving the matrix equations contributes significantly to the cost of Algorithm 8.

5.1.2 Error bounds for Approximate Balanced Truncation

Besides the balanced truncation error bound (5.4) we introduce further errors using the $\mathcal{H}$-matrix format and the corresponding approximate arithmetic. Errors resulting from using the formatted arithmetic during the calculation can be controlled by choosing the parameter for the adaptive rank choice accordingly. As balanced truncation is actually applied to

$$G_{\mathcal{H}}(s) := C(sI - A_{\mathcal{H}})^{-1}B + D,$$

we analyze the influence of the $\mathcal{H}$-matrix error introduced by the approximation of A in $\mathcal{H}$-matrix format in the following. We ignore the influence of rounding errors as they are expected to be negligible compared to the other error sources. For simplifying the notation, we denote the $\mathcal{H}_\infty$-norm of the TFM by $\|\cdot\|_{\mathcal{H}_\infty}$ instead of $\|\cdot\|_{\mathcal{H}_\infty(\mathbb{C}^+ \to \mathbb{C}^{p\times m})}$ throughout this section. We can split the approximation error into two parts using the triangle inequality

$$\|G - \hat{G}\|_{\mathcal{H}_\infty} \leq \|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} + \|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}, \tag{5.15}$$

where the second part can essentially be estimated by the balanced truncation error bound (5.4). In the following, we derive some bounds for the first term which accounts for the $\mathcal{H}$-matrix approximation error. We note that the following results are related to the perturbation theory for transfer functions derived in [153] and can partially be obtained as special cases of error bounds given there.

First, we note the identity

$$C(\jmath\omega I - A)^{-1}B - C(\jmath\omega I - A_{\mathcal{H}})^{-1}B = C[(\jmath\omega I - A)^{-1}(A - A_{\mathcal{H}})(\jmath\omega I - A_{\mathcal{H}})^{-1}]B.$$

Hence, the error between G and $G_{\mathcal{H}}$ can be expressed as

$$\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} = \sup_{\omega\in\mathbb{R}} \|C\left[(\jmath\omega I - A)^{-1}(A - A_{\mathcal{H}})(\jmath\omega I - A_{\mathcal{H}})^{-1}\right]B\|_2.$$

Thus

$$\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} \leq \|C\|_2\|B\|_2\|A - A_{\mathcal{H}}\|_2 \sup_{\omega\in\mathbb{R}} \|(\jmath\omega I - A)^{-1}\|_2 \sup_{\omega\in\mathbb{R}} \|(\jmath\omega I - A_{\mathcal{H}})^{-1}\|_2. \tag{5.16}$$

The latter two terms in the estimate are the reciprocals of the complex stability radii [98, Theorem 5.3.36] of A and $A_{\mathcal{H}}$, respectively. Several algorithms have been proposed for computing the complex stability radius, see references in [98, Section 5.3.9]. As in our application $A_{\mathcal{H}}$ comes from the $\mathcal{H}$-matrix approximation of some elliptic operator, we provide some specific bounds for matrices obtained in these situations.

Theorem 5.1.2 *Let the distance between A and its $\mathcal{H}$-matrix approximant be defined by*

$$\eta := \|A_{\mathcal{H}} - A\|_2.$$

Let A and $A_{\mathcal{H}}$ be stable and assume that both matrices are diagonalizable so that

$$T^{-1}AT = \operatorname{diag}(\lambda_1, \ldots, \lambda_n), \quad \tilde{T}^{-1}(A_{\mathcal{H}})\tilde{T} = \operatorname{diag}(\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n).$$

Furthermore, assume that

$$\operatorname{cond}_2(T)\,\eta \leq \min_{i=1,\ldots,n} |Re(\lambda_i(A))|. \tag{5.17}$$

Then the $\mathcal{H}_\infty$-norm of the corresponding error system $G - G_{\mathcal{H}}$ is bounded by

$$\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} \leq \|C\|_2\|B\|_2 \operatorname{cond}_2(T) \operatorname{cond}_2\left(\tilde{T}\right) \frac{1}{\min\limits_{i=1,\ldots,n} |Re(\lambda_i(A))|^2}\mathcal{O}(\eta). \tag{5.18}$$

Proof: Using the notation

$$D := \operatorname{diag}(\lambda_1, \ldots, \lambda_n), \quad \tilde{D} := \operatorname{diag}(\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n),$$

setting

$$\mu = \min_{i=1,\ldots,n} |\mathrm{Re}(\lambda_i(A))|, \quad \tilde{\mu} = \min_{i=1,\ldots,n} |\mathrm{Re}(\lambda_i(A_{\mathcal{H}}))|,$$

and invoking (5.16) yields, by simple calculations, the following bounds:

$$\begin{aligned}
&\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} \\
&\qquad \leq \quad \eta\,\|C\|_2\|B\|_2 \operatorname{cond}_2(T) \operatorname{cond}_2\left(\tilde{T}\right) \sup_{\omega\in\mathbb{R}} \|(\jmath\omega I - D)^{-1}\|_2 \\
&\qquad\qquad \sup_{\omega\in\mathbb{R}} \|(\jmath\omega I - \tilde{D})^{-1}\|_2 \\
&\qquad \overset{(*)}{=} \quad \eta\,\|C\|_2\|B\|_2 \operatorname{cond}_2(T) \operatorname{cond}_2\left(\tilde{T}\right) \frac{1}{\mu\,\tilde{\mu}} \\
&\qquad \overset{(**)}{\leq} \quad \eta\,\|C\|_2\|B\|_2 \operatorname{cond}_2(T) \operatorname{cond}_2\left(\tilde{T}\right) \frac{1}{\mu(\mu - \operatorname{cond}_2(T)\,\eta)} \\
&\qquad \overset{(***)}{\leq} \quad \eta\,\|C\|_2\|B\|_2 \operatorname{cond}_2(T) \operatorname{cond}_2\left(\tilde{T}\right) \left(\frac{1}{\mu^2} + \frac{1}{\mu^3}\mathcal{O}(\eta)\right).
\end{aligned}$$

The identity $(*)$ follows from the observation that the maximum of $1/|\jmath\omega - \lambda|$ over the imaginary axis is taken for the eigenvalue closest to the imaginary axis, that is the eigenvalue with minimal absolute value of the real part. The estimate in $(**)$ is a consequence of the Bauer-Fike theorem, see, e.g., [74, Theorem 7.2.2]. Due to (5.17) we can apply the geometric series to obtain $(***)$. □

For unitarily diagonalizable A as obtained, e.g., from a finite-differences discretization of a self-adjoint elliptic operator, the error bound (5.18) simplifies.

Corollary 5.1.3 *With the same assumptions as in Theorem 5.1.2, and assuming additionally that A and $A_{\mathcal{H}}$ are unitarily diagonalizable by*

$$U^H A U = \operatorname{diag}(\lambda_1, \ldots, \lambda_n), \quad \tilde{U}^H A_{\mathcal{H}} \tilde{U} = \operatorname{diag}(\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n),$$

we obtain the error bound

$$\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} \leq \|C\|_2 \|B\|_2 \frac{1}{\min\limits_{i=1,\ldots,n} |Re(\lambda_i(A))|^2} \mathcal{O}(\eta).$$

□

Thus, for a symmetric, negative-definite A with spectrum

$$\lambda_n \leq \ldots \leq \lambda_1 < 0 \tag{5.19}$$

and symmetric, negative-definite approximation $A_{\mathcal{H}}$ we get the error bound

$$\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty} \leq \frac{1}{\lambda_1^2} \|C\|_2 \|B\|_2 \mathcal{O}(\eta). \tag{5.20}$$

Remark 5.1.4 As described in our Example 4.1.13, using the Cholesky factor of the mass matrix in a finite element discretization of a self-adjoint spatial differential operator, the transformed system has a symmetric, negative-definite state matrix A. For these systems the assumptions of Corollary 5.1.3 are fulfilled.

□

We can now state our main result of this section which combines the errors due to the $\mathcal{H}$-matrix approximation and balanced truncation.

Theorem 5.1.5 *With $\hat{G}$ as TFM associated to the reduced-order system (5.1) obtained by applying balanced truncation to $G_{\mathcal{H}}$ and the assumptions of Theorem 5.1.2, we obtain for the whole approximation error (5.15)*

$$\begin{aligned} \|G - \hat{G}\|_{\mathcal{H}_\infty} \quad \leq \quad & \|C\|_2 \|B\|_2 \operatorname{cond}_2(T) \operatorname{cond}_2\left(\tilde{T}\right) \frac{1}{\min\limits_{i=1,\ldots,n} |Re(\lambda_i(A))|^2} \mathcal{O}(\eta) \\ & + 2 \left(\sum_{j=r+1}^{n} \sigma_j \right). \end{aligned}$$

The bound simplifies for the practically relevant case of symmetric, negative definite matrices A and $A_{\mathcal{H}}$ with ordered real eigenvalues $\lambda_i \in \Lambda(A)$ as in (5.19) to

$$\|G - \hat{G}\|_{\mathcal{H}_\infty} \leq \frac{1}{\lambda_1^2} \|C\|_2 \|B\|_2 \mathcal{O}(\eta) + 2 \left(\sum_{j=r+1}^{n} \sigma_j \right).$$

All error bounds derived in this section are of merely qualitative nature and suggest to choose the tolerance for the $\mathcal{H}$-matrix approximation small enough to compensate for possible error amplification due to eigenvalues close to the imaginary axis. Moreover, if the error bound (5.4) is used to adapt the reduced order to a desired accuracy tolerance, then the tolerance should be chosen according to the estimated size of the first part of the error such that both error parts are balanced.

In the next section, we will show how the approximation errors actually behave in numerical computations.

5.1.3 Numerical Results

Before we test the developed algorithm, we consider how to measure the accuracy of the resulting reduced-order system in practice. Note that for large-scale, dense problems we can only compute the second part in (5.15), i.e., $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$, of the error between original and reduced-order system. This is caused by the quadratic memory requirements for the solution of a linear system of equations with the original dense matrix $(\jmath\omega I - A)$ in order to compute $G(\jmath\omega)$ which becomes unfeasible for large-scale systems. For sparse systems of large dimensions we can also compute the $\mathcal{H}$-matrix approximation error $\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty}$ and the error between the original and the reduced-order system $\|G - \hat{G}\|_{\mathcal{H}_\infty}$. Recall from Section 2.1.3 that the $\mathcal{H}_\infty$-norm of an error system, e.g. of $G - \hat{G}$, is given by the supremum of the 2-induced norm on the imaginary axis, i.e.,

$$\|G - \hat{G}\|_{\mathcal{H}_\infty(\mathbb{C}^+ \to \mathbb{C}^{p\times m})} = \sup_{\omega\in\mathbb{R}} \sigma_{\max}(G(\jmath\omega) - \hat{G}(\jmath\omega)).$$

This norm can be approximated by computing $\sigma_{\max}(G(\jmath\omega) - \hat{G}(\jmath\omega))$ at some points in a range of interest, $\omega_k \in (0, \omega_{\max}]$. Note that for SISO systems the computation simplifies to $|G(\jmath\omega_k) - \hat{G}(\jmath\omega_k)|$. If the errors were plotted against the frequency, using logarithmic scales for both axes it is the usual Bode magnitude plot, see Example 2.1.19.

Remark 5.1.6 (Computing Frequency Response Errors in $\mathcal{H}$-Matrix Arithmetic)

For computing a bound for the latter part $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ of the error estimate (5.15) we have to note that the $\mathcal{H}$-matrix format is defined only for real-valued matrices. So we have to compute the frequency response of the complex transfer function $G_{\mathcal{H}}$ separately for the real and for the imaginary part. We will treat the continuous-time case in more detail. We evaluate $G_{\mathcal{H}}$ at a fixed frequency ω_k. For the complex-valued matrix in the definition of the transfer function we consider a splitting into real part X_{Re} and imaginary part X_{Im}:

$$G_{\mathcal{H}}(\jmath\omega_k) = C\left(\jmath\omega_k I - A_{\mathcal{H}}\right)^{-1} B + D = C(X_{\mathrm{Re}} + \jmath\, X_{\mathrm{Im}}) + D.$$

Since B is real-valued we obtain a system of equations for the unknowns X_{Re} and X_{Im}

$$\begin{array}{rcrcl} -A_{\mathcal{H}}X_{\mathrm{Re}} & - & \omega_k X_{\mathrm{Im}} & = & B, \\ \omega_k X_{\mathrm{Re}} & - & A_{\mathcal{H}}X_{\mathrm{Im}} & = & 0, \end{array}$$

and by some simple calculations the following solution formulas:

$$\begin{aligned} X_{\mathrm{Re}} &= -(A_{\mathcal{H}}^2 + \omega_k^2 I)^{-1} A_{\mathcal{H}} B, \\ X_{\mathrm{Im}} &= -\omega_k (A_{\mathcal{H}}^2 + \omega_k^2 I)^{-1} B. \end{aligned}$$

The norm of the error system can now be approximated using formatted arithmetics:

$$\begin{aligned} \|G_{\mathcal{H}}(\jmath\omega_k) - \hat{G}(\jmath\omega_k)\|_2 &= \sigma_{\max}(G_{\mathcal{H}}(\jmath\omega_k) - \hat{G}(\jmath\omega_k)) \\ &= \sigma_{\max}(C(X_{\mathrm{Re}} + \jmath X_{\mathrm{Im}}) - \hat{C}(\jmath\omega I - \hat{A})^{-1}\hat{B}) \\ &= \sigma_{\max}(C\left[-(A_{\mathcal{H}} \odot A_{\mathcal{H}} \oplus \omega_k^2 I)_{\mathcal{H}}^{-1} A_{\mathcal{H}} B - \right. \\ &\qquad \left. \jmath\omega_k (A_{\mathcal{H}} \odot A_{\mathcal{H}} \oplus \omega_k^2 I)_{\mathcal{H}}^{-1} B\right] - \hat{C}(\jmath\omega I - \hat{A})^{-1}\hat{B}). \end{aligned}$$

For discrete-time systems the absolute error is computed as the maximum singular value of the error system $G_{\mathcal{H}}(z) - \hat{G}(z)$ for $z = e^{\jmath\omega_k T_s}$ and sampling time T_s:

$$\|G_{\mathcal{H}}(e^{\jmath\omega_k T_s}) - \hat{G}(e^{\jmath\omega_k T_s})\|_2 = \sigma_{\max}(G_{\mathcal{H}}(e^{\jmath\omega_k T_s}) - \hat{G}(e^{\jmath\omega_k T_s})),$$

at some fixed frequencies $\omega_k \in (0, \omega_N]$ where $\omega_N = \pi/T_s$ is the so-called *Nyquist frequency*, see, e.g. [130]. To simplify the expression we restrict to the frequency response of $G_{\mathcal{H}}$ using $\omega := \omega_k T_s$:

$$\begin{aligned} \|G_{\mathcal{H}}(e^{\jmath\omega})\|_2 &= \sigma_{\max}\Big(C\Big[\big(\cos\omega I \ominus A_{\mathcal{H}} \oplus \sin^2\omega (\cos\omega I \ominus A_{\mathcal{H}})_{\mathcal{H}}^{-1}\big)_{\mathcal{H}}^{-1} \\ &\qquad -\jmath\,(\cos\omega I \ominus A_{\mathcal{H}})_{\mathcal{H}}^{-1} \sin\omega \big(\cos\omega I \ominus A_{\mathcal{H}} \\ &\qquad \oplus \sin^2\omega\,(\cos\omega I \ominus A_{\mathcal{H}})_{\mathcal{H}}^{-1}\big)_{\mathcal{H}}^{-1}\Big] B\Big). \end{aligned}$$

Note that in order to obtain accurate results for these norm approximations we set the accuracy in the adaptive arithmetic to 10^{-16} in all calculations described in this remark. This is very time demanding and therefore, the error systems are evaluated only at 20 to 30 frequencies in the following examples. □

As exemplary systems for model order reduction of continuous-time systems we consider the examples from Section 4.2.2. We apply the $\mathcal{H}$-matrix based BT method (Algorithm 8) to systems $\Sigma(A_{\mathcal{H}}, B, C)$ of order $n = 16,384$ in all examples based on the control problem for the two-dimensional heat equation 4.1.13. The reduced order r is determined by the threshold tol for the approximation quality. With the computable part of the error estimate δ (defined in (5.14)), r is chosen as minimal integer such that $\delta \leq \mathrm{tol}$. In all examples, the frequency response errors between $G_{\mathcal{H}}$ and $\hat{G}$ are approximated by use of the formatted $\mathcal{H}$-matrix arithmetic as described in Remark 5.1.6. The pointwise absolute values of the error systems are computed at 20 fixed frequencies ω_k from $10^{-4}, \ldots, 10^6$ in logarithmic scale in most of the examples. If not otherwise stated, we choose a fixed parameter choice of $\tau = \epsilon = 10^{-6}$ in Algorithm 5 for the solution of the coupled Lyapunov equations in the first stage of the approximate BT method. This choice is motivated by the numerical results in Section 4.2.2 and by the discussion in Section 5.1.1 which suggests a choice of $\tau \sim \epsilon$ to bound errors as introduced by the use of approximate Gramians.

First, we apply Algorithm 8 to the LTI system resulting from the two-dimensional heat equation with $a(\cdot) \equiv 1.0$ in Example 4.2.5. We conclude from Table 4.5 that by $\tau = \epsilon = 10^{-6}$, the relative errors between the exact and the computed Gramians ($\tilde{\mathcal{P}} := \tilde{S}\tilde{S}^T,\ \tilde{\mathcal{Q}} := \tilde{R}\tilde{R}^T$) are approximately

$$\frac{\|\mathcal{P} - \tilde{\mathcal{P}}\|_F}{\|\mathcal{P}\|_F} \sim \frac{\|\mathcal{Q} - \tilde{\mathcal{Q}}\|_F}{\|\mathcal{Q}\|_F} \sim 10^{-7}.$$

The reduced order is determined by $r = 4$ for the threshold $\text{tol} = 10^{-4}$. That is, the HSVs decay very rapidly, the approximate error bound is $\delta = 4.3 \times 10^{-5}$. The frequency response errors between $G_{\mathcal{H}}$ and $\hat{G}$ are shown in Figure 5.1. We observe a typical good approximation of the BT method for larger frequencies as

$$\lim_{\omega \to \infty} (G(\jmath\omega) - \hat{G}(\jmath\omega)) = D - \hat{D} = 0.$$

We also depict the errors between the original (full) and the reduced-order system $G - \hat{G}$ to demonstrate the reliability of our approach. Note that there is no visible difference between the two error plots and both curves satisfy the approximate error bound δ. Thus, other error sources using approximate balanced truncation seem to be negligible. In the lower plot we add the difference in the frequency response of the unreduced systems $\Sigma(A, B, C)$ and $\Sigma(A_{\mathcal{H}}, B, C)$. Thus, all errors in the estimate (5.15) are plotted. It is seen that both parts $\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty}$ and $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ on the right-hand side of the error estimate (5.15) are nearly balanced. If we choose a larger value of 10^{-4} for the accuracy ϵ of the adaptive arithmetic, the approximation error $\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty}$ increases, see the middle plot in Figure 5.2. It follows that for smaller frequencies the difference between original and reduced-order system is slightly larger than the computed error estimate but still smaller than the desired tolerance of 10^{-4} (see upper plot in Figure 5.2). Note that an increase of ϵ reduces the required storage as well as the computational time (from 4508 seconds for $\epsilon = 10^{-6}$ to 2151 seconds for $\epsilon = 10^{-4}$) of the approximate BT method.

In the lower plot in Figure 5.2 we additionally depict errors from a reduced-order system of size $r = 7$ obtained by a choice of $\text{tol} = 10^{-6}$. It is observed that the error bound estimate $\delta = 1.5 \times 10^{-7}$ is satisfied by $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ as well as by $\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty}$.

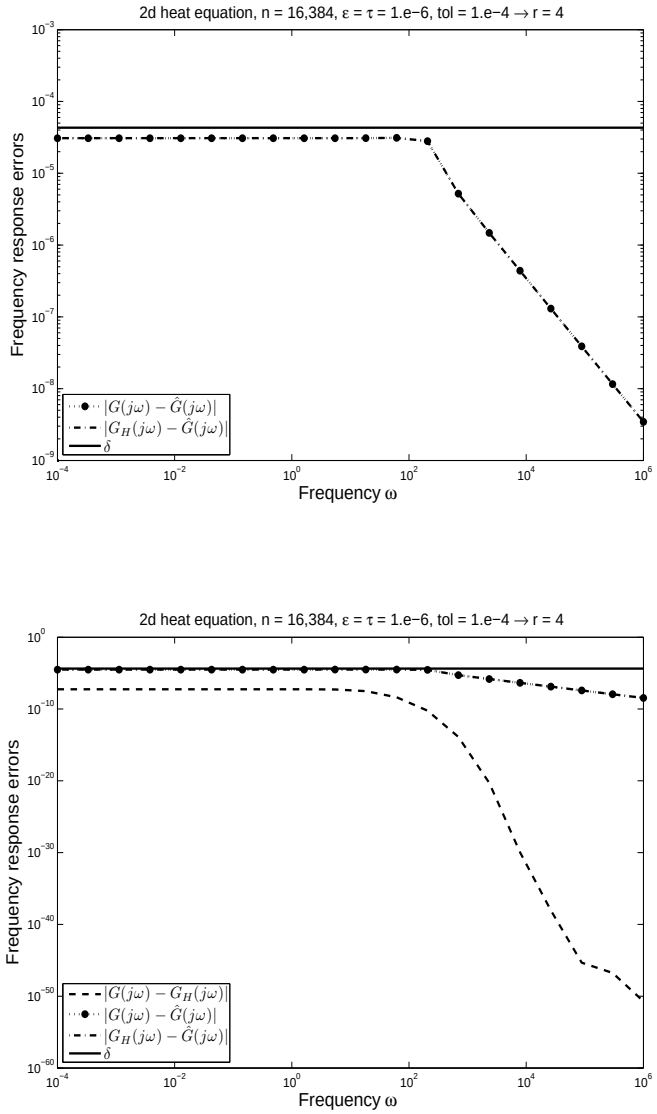


Figure 5.1: Frequency response errors for the two-dimensional heat equation using approximate BT as described in Algorithm 8.

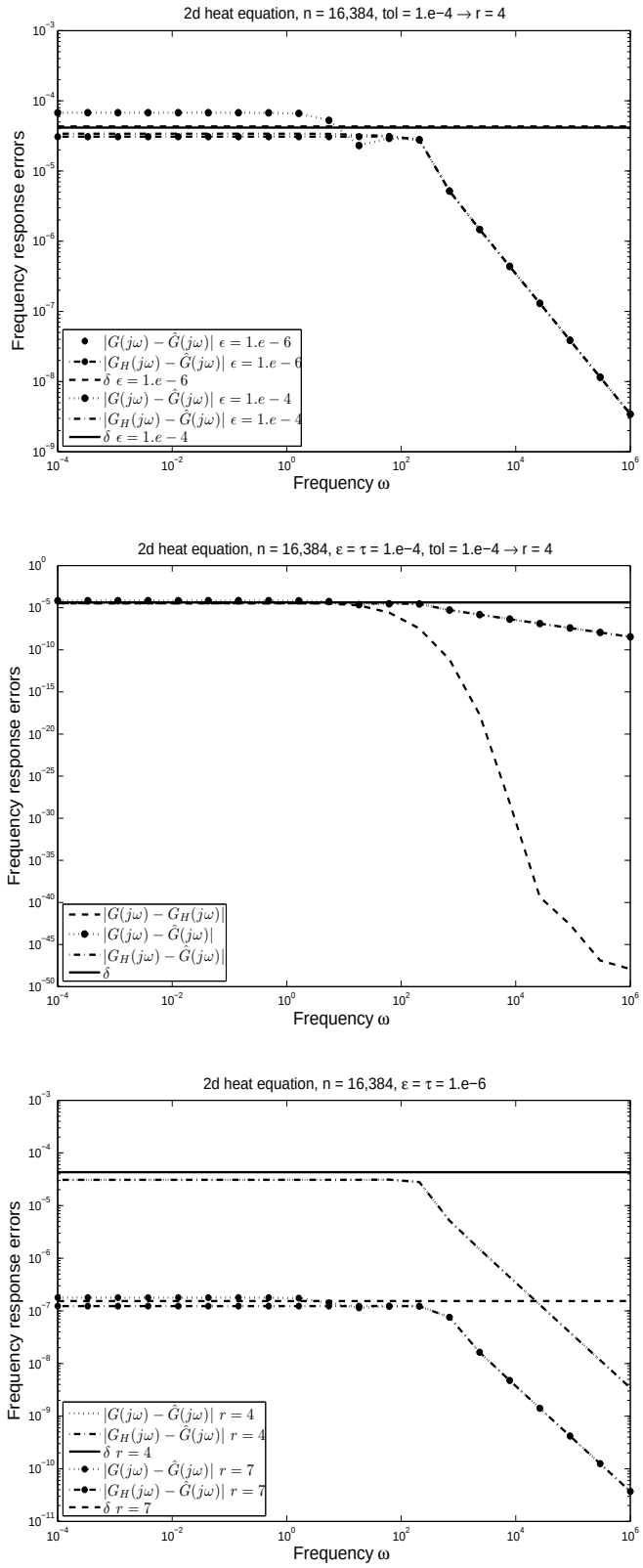


Figure 5.2: Frequency response errors for the two-dimensional heat equation using approximate BT as described in Algorithm 8.

Next, BT is applied to reduce the dimension of the heat equation with varying diffusion as introduced in Example 4.2.6. Setting tol $= 10^{-4}$ a very small reduced order of $r = 3$ is determined. The frequency response errors are plotted in Figure 5.3 and again, we observe good matching at high frequencies for the reduced-order model. The results fulfill the approximate BT error bound $\delta = 8.6 \times 10^{-5}$, the difference between $\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty}$ and $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ is again negligible. Thus, using approximate Gramians does not contribute much to the errors between original and reduced-order system.

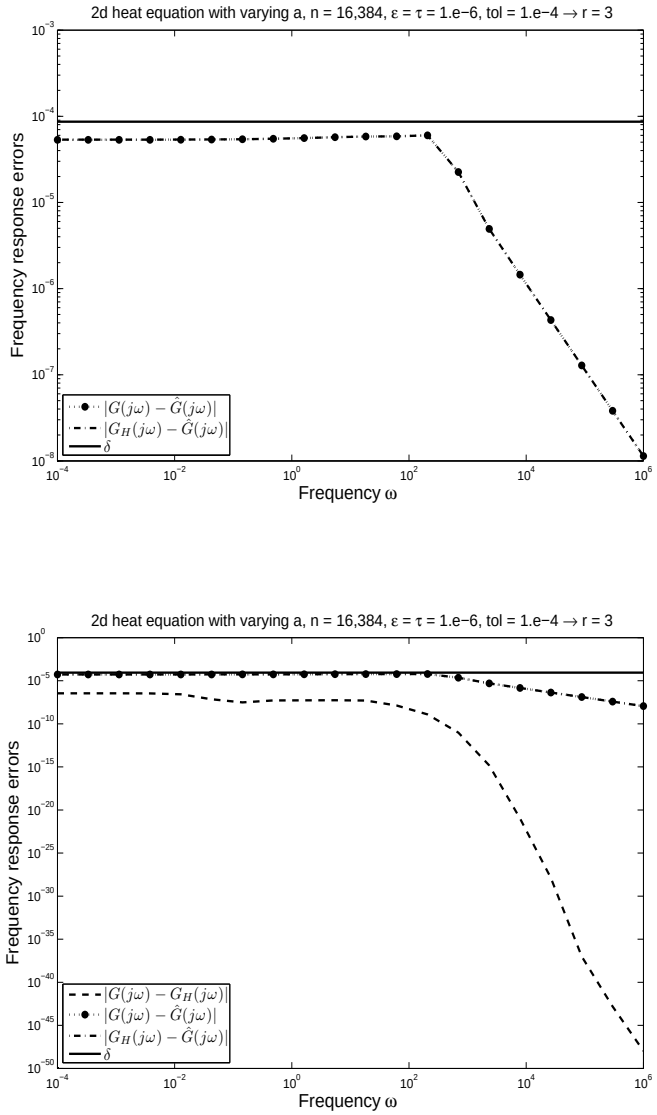


Figure 5.3: Frequency response errors for the two-dimensional heat equation with varying diffusion using approximate BT as described in Algorithm 8.

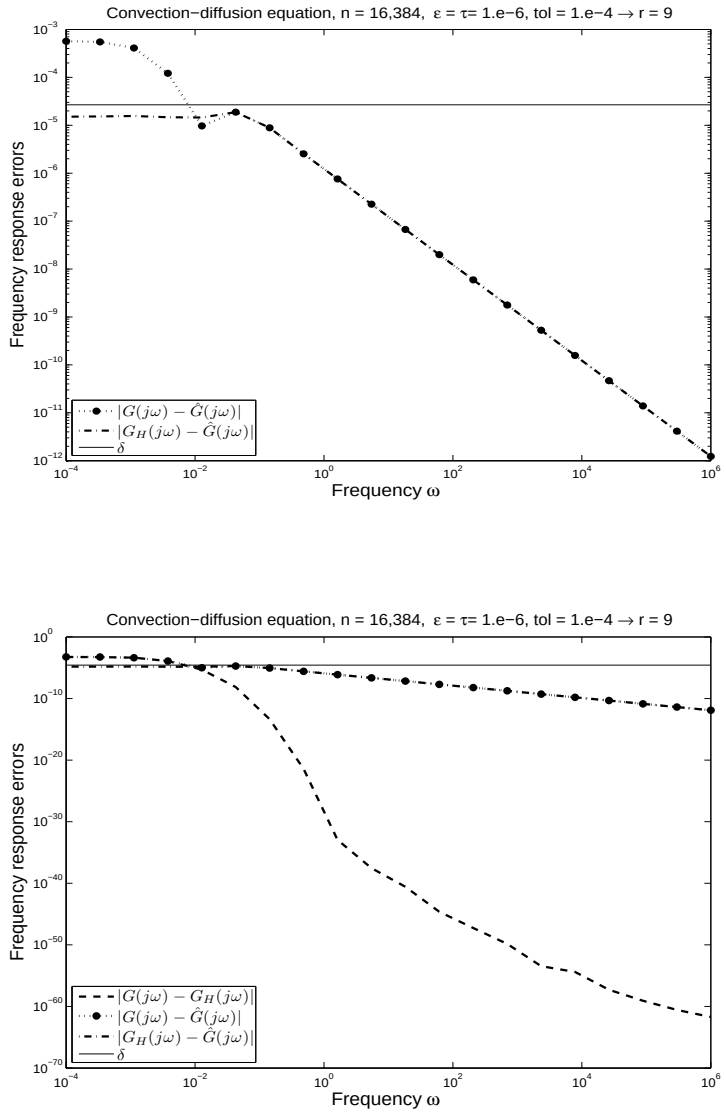


Figure 5.4: Frequency response errors for the convection-diffusion equation using approximate BT as described in Algorithm 8 with $\epsilon = \tau = 10^{-6}$.

We apply the approximate BT method to the convection-diffusion equation studied in Example 4.2.7. For the constant diffusion coefficient $a(\cdot) \equiv 10^{-4}$, the convective term is dominant and the eigenvalues of A are close to the imaginary axis, i.e. $\min_{i=1,\ldots,n} |\mathrm{Re}(\lambda_i(A))| \approx 2. \times 10^{-3}$. Furthermore, the condition number of T is much larger than 1, an estimate for $n = 4096$ yields: $\mathrm{cond}_2(T) \approx 4.5 \times 10^3$. For tol $= 10^{-4}$ a reduced-order system of size $r = 9$ and the error estimate $\delta = 2.67 \times 10^{-5}$ are computed. We observe in Figure 5.4 that the error between the reduced-order system and $\Sigma(A_{\mathcal{H}}, B, C)$ satisfies the error estimate. The difference between the original system $\Sigma(A, B, C)$ and $\Sigma(A_{\mathcal{H}}, B, C)$ is larger than δ for frequencies smaller than 10^{-2}, see the lower plot in Figure 5.4. Note that

the condition number of T as well as the reciprocal of the square of the critical eigenvalue (that one closest to the imaginary axis) amplify the error between G and $G_{\mathcal{H}}$ in (5.18), see Theorem 5.1.2. The error system $G - \hat{G}$ is influenced by this error, see (5.15). By a choice of $\epsilon = 10^{-8}$, the error $\|G - G_{\mathcal{H}}\|_{\mathcal{H}_\infty}$ decreases and the error between original and reduced-order system satisfies the computed error estimate $\delta = 3.29 \times 10^{-5}$, see both plots in Figure 5.5.

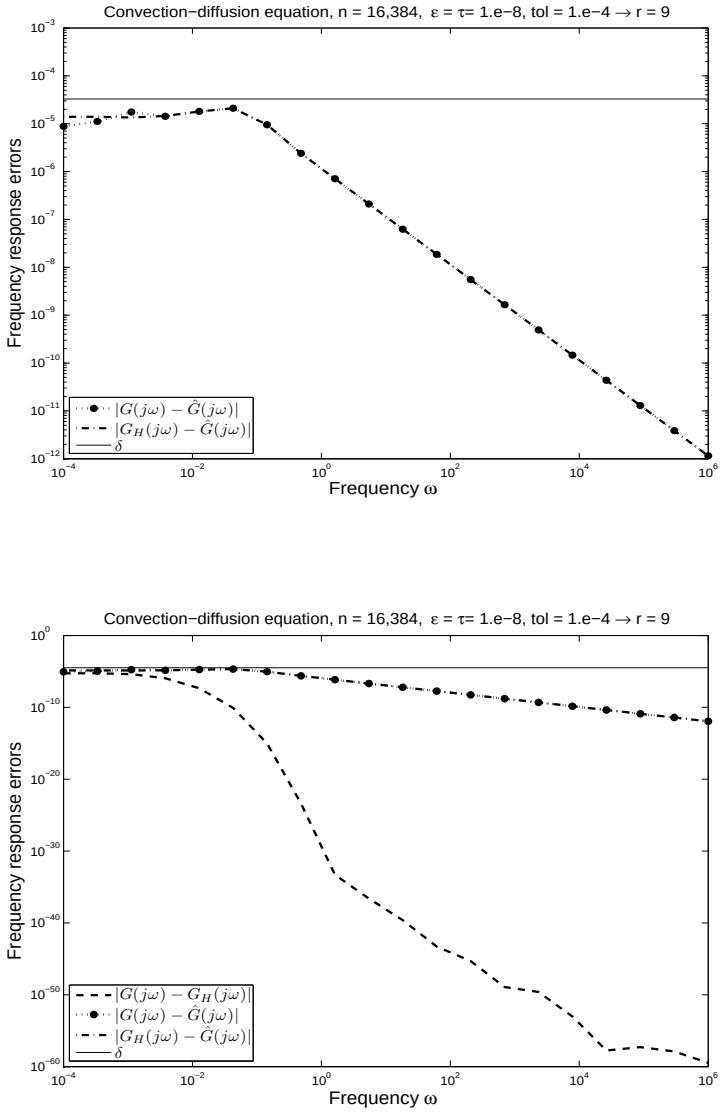


Figure 5.5: Frequency response errors for the convection-diffusion equation using approximate BT as described in Algorithm 8 with $\epsilon = \tau = 10^{-8}$.

As next example, we consider a large-scale, dense problem from a boundary element discretization of the Laplacian on a three-dimensional sphere as introduced in Example 4.2.8, additionally setting $C^T = B$ to obtain an output equation. We choose a problem size of $n = 32,768$ and depict the frequency response errors $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ for the approximate BT method in a frequency range $10^{-6}, \ldots, 10^{6}$ in Figure 5.6. Note that we are not able to compute the errors between the original and the reduced-order system for this example. For tol $= 10^{-4}$ we determine the order $r = 10$ and the approximate error bound $\delta = 4.82 \times 10^{-5}$. We observe a good approximation quality of the reduced-order system. We would like to emphasize that in this example, we have computed a very small reduced-order model for a fairly large LTI system. In particular, here A is a dense $32,768 \times 32,768$ matrix. This becomes only possible using the data-sparse solver in the approximate BT method. Also note that if the same problem were discretized by FEM with the same accuracy, a system of order $n \approx 6,000,000$ would result! It should be stressed that the storage requirements for the original (dense) system $\Sigma(A, B, C)$ are 8192.5 MB instead of 377.5 MB to store $\Sigma(A_{\mathcal{H}}, B, C)$ and 0.937 KB memory to store the system $\Sigma(\hat{A}, \hat{B}, \hat{C})$ of order 10.

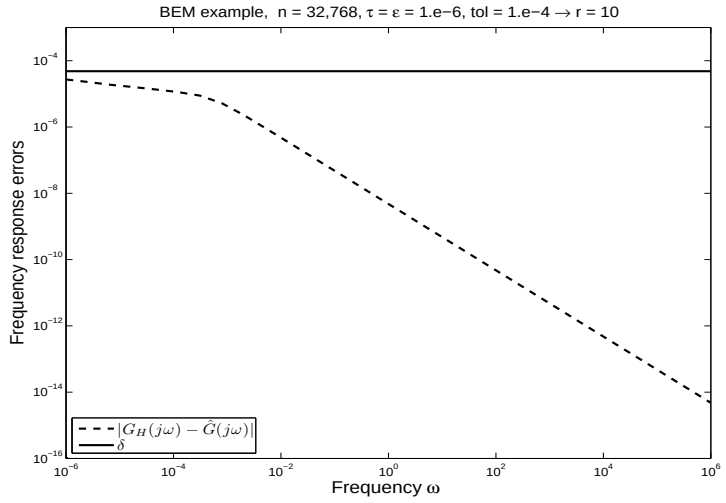


Figure 5.6: Frequency response errors for the BEM example using approximate BT as described in Algorithm 8.

The approximate BT for discrete-time systems, involving the solution of two Stein equations by Algorithm 7 as first computational step, is tested by the exemplary LTI system in Section 4.4.2. The output is described by an observation in a small subdomain as in Example 4.1.13. We set the order of the system to $n = 16,384$, choose the threshold tol $= 10^{-4}$ for the error estimate and use a fixed parameter setting of $\tau = \epsilon = 10^{-6}$. We compute the absolute errors at 30 frequencies ω_k in logarithmic scale with $\omega_k \in [0, \omega_N]$. The absolute errors of the transfer functions for the original full system and the reduced-order model are shown in Figure 5.7. The reduced-order system of order $r = 4$

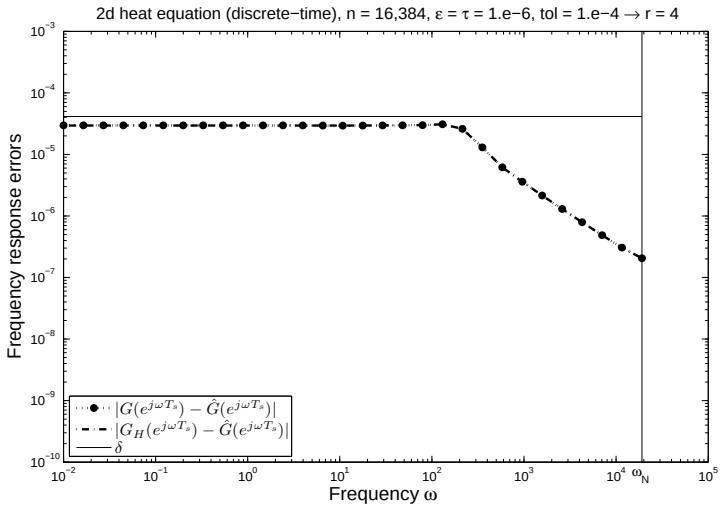


Figure 5.7: Frequency response errors for the discrete-time example using approximate BT as described in Algorithm 8.

matches both systems $\Sigma(A,\, B,\, C)$ and $\Sigma(A_{\mathcal{H}},\, B,\, C)$ very well and satisfies the computed estimate $\delta = 8.39 \times 10^{-5}$.

5.2 Model Reduction with Singular Perturbation Approximation

A reduced-order model with perfect matching of the transfer function G at $\omega = 0$ is required in many applications. In state space this corresponds to a zero steady-state error. Zero steady-state errors can be obtained by a *singular perturbation approximation* to the original system [122, 165], also called *balanced residualization*. It is assumed that the realization of the system (2.1) is minimal (otherwise balanced truncation is used to reduce the order to the McMillan degree of the system) and balanced. Then, in the continuous-time setting, the following partitioned representation is considered

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u(t), \\ y(t) &= \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + D\, u(t), \end{aligned}$$

where $A_{11} \in \mathbb{R}^{r\times r}$, $B_1 \in \mathbb{R}^{r\times m}$, $C_1 \in \mathbb{R}^{p\times r}$ and r is the desired reduced order. The dynamics of the faster state variables x_2 are neglected by setting $\dot{x}_2(t) = 0$ and assuming A_{22} to be nonsingular, and a reduced-order model is obtained as in (5.1) with

$$\begin{aligned} \hat{A} &:= A_{11} - A_{12}A_{22}^{-1}A_{21}, & \hat{B} &:= B_1 - A_{12}A_{22}^{-1}B_2, \\ \hat{C} &:= C_1 - C_2A_{22}^{-1}A_{21}, & \hat{D} &:= D - C_2A_{22}^{-1}B_2. \end{aligned} \tag{5.21}$$

The balanced truncation error bound (5.4) holds as well and the SPA reduced-order model additionally satisifies $\hat{G}(0) = G(0)$ and provides a good approximation at low frequencies.

For discrete-time systems, the formulas

$$\begin{array}{rclrcl} \hat{A} & := & A_{11} + A_{12}(I - A_{22})^{-1}A_{21}, & \hat{B} & := & B_1 + A_{12}(I - A_{22})^{-1}B_2, \\ \hat{C} & := & C_1 + C_2(I - A_{22})^{-1}A_{21}, & \hat{D} & := & D + C_2(I - A_{22})^{-1}B_2, \end{array} \tag{5.22}$$

yield an SPA, where the resulting reduced-order system is stable and balanced and its TFM fulfills $\hat{G}(e^{\jmath \cdot 0}) = \hat{G}(1) = G(1) = G(e^{\jmath \cdot 0})$ [131, Section 1.9].

5.2.1 Approximate Singular Perturbation Approximation

In Algorithm 9 the $\mathcal{H}$-matrix based SPA method is presented. For the computation of a balanced and minimal realization of (2.1) (and of (2.23), in discrete-time) we use the approximate BT method described in Algorithm 8. The McMillan degree is derived by use of the approximate HSVs $\{\sigma_1, \ldots, \sigma_{\tilde{n}}\}$ with $\tilde{n} = \min\{n_\tau(\mathcal{P}), n_\tau(\mathcal{Q})\}$ which are computed by the economy-size SVD of $\tilde{S}^T\tilde{R}$ in (5.10). Using a threshold τ, the approximate McMillan degree $\hat{n}$ is determined by

$$\sigma_{\hat{n}+1} < \sigma_1 \cdot \tau \leq \sigma_{\hat{n}}.$$

The system of "minimal" order $\hat{n}$ is derived by applying the projection matrices (5.11) as described in (5.3). Using (5.7), the reduced order r of the system is determined by the HSVs $\{\sigma_1, \ldots, \sigma_{\hat{n}}\}$ of the minimal realization. The SPA reduced-order model is computed by (5.21) for continuous-time systems and (5.22) for discrete-time systems.

5.2.2 Numerical Results

We use the same examples and parameter settings as in Section 5.1.3. For determining the numerical McMillan degree of the LTI system in the approximate SPA method we use a threshold of size 10^{-14} in all our simulations. Note that by step 1 and 2 in Algorithm 9, the computed HSVs and therefore also the reduced order and the approximate error bound are the same as in the corresponding examples in Section 5.1.3 using the approximate BT method. The frequency response errors between the approximated and the reduced-order systems $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ as well as between the original systems $\Sigma(A, B, C)$ and $\Sigma(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ are depicted in Figures 5.8, 5.9, 5.10 and 5.11. We observe the typical good approximation at low frequencies for the SPA reduced-order models. In all examples based on the heat equation, the visible differences between $\|G - \hat{G}\|_{\mathcal{H}_\infty}$ and $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ for smaller frequencies are caused by the accuracy of $\epsilon = 10^{-6}$ in the adaptive arithmetic and only show up if the errors are smaller than 10^{-6}. For the non-symmetric convection-diffusion example, the error between original and reduced-order system (computed with $\epsilon = 10^{-8}$) is slightly larger for frequencies smaller than 10^{-2}. As analyzed in Section 5.1.2 a large condition number of T and eigenvalues close to the imaginary axis influence the error between $\Sigma(A, B, C)$ and $\Sigma(A_{\mathcal{H}}, B, C)$. This causes larger errors between $\Sigma(A, B, C)$ and $\Sigma(\hat{A}, \hat{B}, \hat{C})$ in Figure 5.10.

Algorithm 9 Approximate SPA for LTI systems (2.1) and (2.23)

INPUT: $A_{\mathcal{H}} \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$; tol, τ, ϵ.
OUTPUT: $\hat{A} \in \mathbb{R}^{r\times r}$, $\hat{B} \in \mathbb{R}^{r\times m}$, $\hat{C} \in \mathbb{R}^{p\times r}$, $\hat{D} \in \mathbb{R}^{p\times m}$; reduced order r, error bound δ.

1: Compute low-rank factors $\tilde{S} \in \mathbb{R}^{n\times n_\tau(\mathcal{P})}$, $\tilde{R} \in \mathbb{R}^{n\times n_\tau(\mathcal{Q})}$ of the system Gramians using Algorithm 5 for continuous-time systems, Algorithm 7 in the discrete-time case.

2: Compute SVD of $\tilde{S}^T\tilde{R}$ ($\tilde{n} := \min\{n_\tau(\mathcal{P}), n_\tau(\mathcal{Q})\}$)

$$\tilde{S}^T\tilde{R} = U\Sigma V^T,$$

with $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_{\tilde{n}})$ and HSVs in decreasing order.
Determine numerical McMillan degree $\hat{n}$ by threshold τ.

3: Compute balanced and minimal realization by (5.3) using projection matrices (5.11).

4: Partition matrices according to reduced order r ($A_{11} \in \mathbb{R}^{r\times r}$), r is determined by tol: $\delta = 2\sum_{j=r+1}^{\tilde{n}} \sigma_j \leq \mathrm{tol}$,

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, C = \begin{bmatrix} C_1 & C_2 \end{bmatrix}.$$

5: Compute SPA reduced-order model $\hat{A}$, $\hat{B}$, $\hat{C}$, $\hat{D}$ with formulas (5.21) for continuous-time and with (5.22) for discrete-time systems.

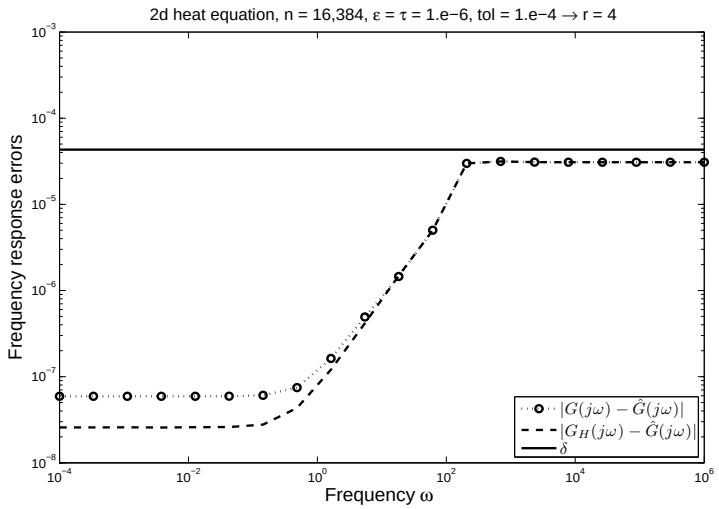


Figure 5.8: Frequency response errors for the two-dimensional heat equation using approximate SPA as described in Algorithm 9.

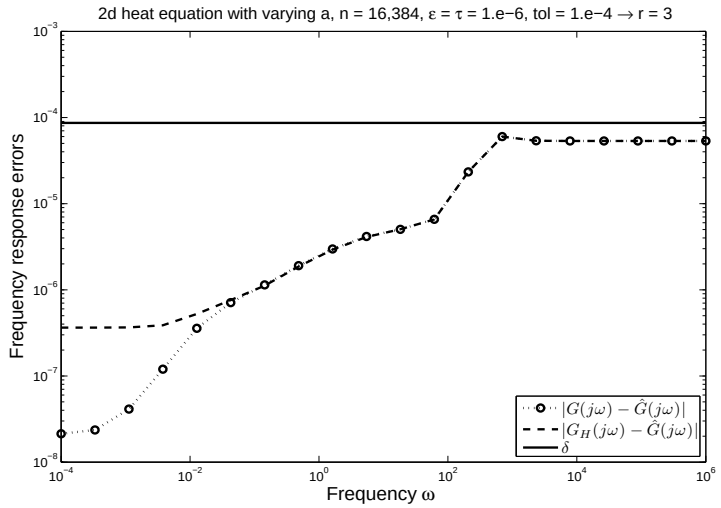


Figure 5.9: Frequency response errors for the two-dimensional heat equation with varying diffusion using approximate SPA as described in Algorithm 9.

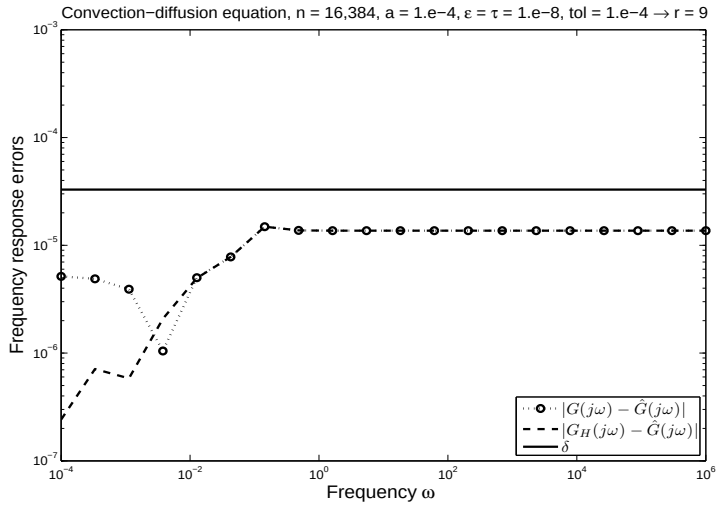


Figure 5.10: Frequency response errors for the convection-diffusion equation using approximate SPA as described in Algorithm 9.

5.3 Cross-Gramian Model Reduction

In 1983 a new system Gramian was defined for stable SISO systems,

$$\mathcal{X} := \int_0^\infty e^{At} BC\, e^{At} dt, \tag{5.23}$$

which contains information on the controllability of the system as well as on the observability [64]. Therefore, the Gramian $\mathcal{X} \in \mathbb{R}^{n \times n}$ is called the *cross-*

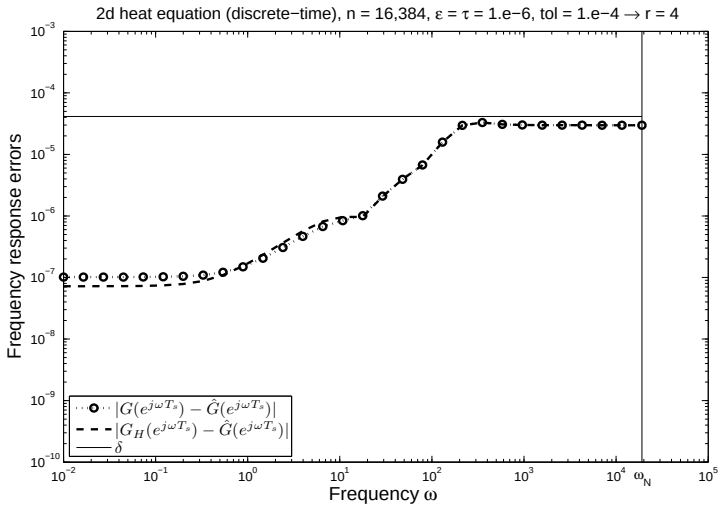


Figure 5.11: Frequency response errors for for discrete-time example using approximate SPA as described in Algorithm 9.

Gramian of the system. The definition was extended to symmetric MIMO systems [65, 116]. Note that a realization $\Sigma(A, B, C, D)$ is called symmetric if the corresponding transfer function $G(\cdot)$ is symmetric. This is trivially the case for systems with $A = A^T$, $B = C^T$. In [65, 116] properties of the cross-Gramian were derived which underline the usefulness of $\mathcal{X}$ for the purpose of model order reduction. The most important result is stated in the following theorem.

Theorem 5.3.1 [64, 65, 116] *For SISO and for symmetric MIMO systems, the cross-Gramian satisfies*

$$\mathcal{X}^2 = \mathcal{P}\mathcal{Q}.$$

That is, we have an equivalence relation to balanced truncation for these systems. Instead of computing the controllability and the observability Gramian it is sufficient to compute the cross-Gramian $\mathcal{X}$. As the cross-Gramian is equivalently given by the solution of the Sylvester equation

$$A\mathcal{X} + \mathcal{X}A + BC = 0, \tag{5.24}$$

we have to solve one Sylvester equation instead of two Lyapunov equations as for balanced truncation model reduction. A further motivation for this approach is given in [156] by the following consistency argument. In balanced truncation the basis sets T_l and T_r for projection are computed by (5.11) from approximations to the exact controllability and observability Gramians. Since both Gramians are approximated separately it can not be ensured that the same basis sets would have been computed by the full system Gramians. In other words, there might be a gap between the approximation errors of $\mathcal{P}$ and $\mathcal{Q}$ which influences the computed reduced-order system in some way. This problem does not occur if we compute projection matrices from an approximation to the cross-Gramian.

The HSVs are given by the magnitude of the eigenvalues of $\mathcal{X}$,

$$\sigma_i = |\lambda_i(\mathcal{X})|, \quad \text{for } i = 1, \ldots, n,$$

and under state space transformation, the eigenvalues are invariant

$$\hat{\mathcal{X}} = T^{-1}\mathcal{X}T.$$

If $\mathcal{X}$ is diagonalizable and T is a balancing transformation then

$$\hat{\mathcal{X}} = \operatorname{diag}(\lambda_1, \cdots, \lambda_n), \quad \text{with } |\lambda_1| \geq \cdots \geq |\lambda_n|.$$

The reduced-order system is simply given by the first r states of the balanced realization. In general, if the system dynamics are projected onto the eigenspaces associated with the largest eigenvalues of $\mathcal{X}$, the reduced-order model has the same properties as in balanced truncation model reduction, i.e. the stability is preserved and a computable global error bound exists.

We shortly review the computational steps in the work of Aldhaheri [1] since the paper initiated the approach considered here. In [1], the left and right eigenspaces of $\mathcal{X}$ corresponding to the eigenvalues of largest magnitude are computed as follows. To compute the dominant right eigenspaces of the cross-Gramian, $\mathcal{X}$ is transformed into an ordered real Schur form,

$$\begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} \mathcal{X} \begin{bmatrix} U_1, & U_2 \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix}, \tag{5.25}$$

where the r eigenvalues with largest magnitude are ordered at the diagonal of the upper triangular block $S_{11} \in \mathbb{R}^{r\times r}$ and $\Lambda(S_{11}) \cap \Lambda(S_{22}) = \emptyset$. The columns in U_1 span the unique right invariant subspace associated with $\Lambda(S_{11})$. We define

$$\begin{bmatrix} U_1, & U_2 \end{bmatrix}^{-1} =: \begin{bmatrix} W_1 \\ W_2 \end{bmatrix},$$

such that the left invariant subspace associated with $\Lambda(S_{11})$ is spanned by the rows of W_1. The Schur form in (5.25) is block diagonalized by computing the solution $X \in \mathbb{R}^{r\times(n-r)}$ of the Sylvester equation

$$S_{11}X - XS_{22} + S_{12} = 0,$$

such that

$$\begin{bmatrix} I_r & -X \\ 0 & I_{n-r} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} \begin{bmatrix} I_r & X \\ 0 & I_{n-r} \end{bmatrix} = \begin{bmatrix} S_{11} & 0 \\ 0 & S_{22} \end{bmatrix}.$$

Then, W_1 can simply be derived by $W_1 = U_1^T - XU_2^T$. The reduced-order system is obtained without computing balancing transformations, applying the transformation matrices $T_l = W_1, \quad T_r = U_1$ as in (5.3) to the system (2.1). Note that the computational complexity restricts the applicability of the method to smaller problem sizes.

5.3.1 Approximate Cross-Gramian Approach

We propose another algorithm for computing projection matrices which is also suitable in the large-scale setting. As discussed in Section 3.1.2 it is often observed that the singular values of $\mathcal{X}$ have an exponential decay. This gave the motivation for computing the cross-Gramian in factored form $\mathcal{X} \approx \tilde{Y}\tilde{Z}$ with $\tilde{Y} \in \mathbb{R}^{n\times n_\tau(\mathcal{X})}$, $\tilde{Z} \in \mathbb{R}^{n_\tau(\mathcal{X})\times n}$ exploiting the expected low numerical rank of $\mathcal{X}$: $n_\tau(\mathcal{X}) \ll n$. We compute the projection matrices T_l and T_r as the left and right dominant invariant subspaces of $\mathcal{X}$ by a left and right eigenvalue decomposition of $\tilde{Y}\tilde{Z}$. We propose a numerical efficient and accurate algorithm for the computation of these subspaces. First, a basis for the right invariant subspace of $\tilde{Y}\tilde{Z}$ corresponding to the r largest eigenvalues is computed,

$$(\tilde{Y}\tilde{Z})\, V_r = V_r \Lambda_1,$$

where $\Lambda_1 = \operatorname{diag}(\lambda_1, \cdots, \lambda_r)$ and the eigenvalues are in descending order (by magnitude). The remaining $n-r$ eigenvalues of $\tilde{Y}\tilde{Z}$ are smaller in magnitude. The columns of $V_r \in \mathbb{R}^{n\times r}$ span the dominant right eigenspaces of $\tilde{Y}\tilde{Z}$. In practice, we compute an eigenvalue decomposition of the "small" matrix product $\tilde{Z}\tilde{Y} \in \mathbb{R}^{n_\tau(\mathcal{X})\times n_\tau(\mathcal{X})}$. The eigenvalue decomposition will be done without explicitly computing the product of the two factors $\tilde{Z}$ and $\tilde{Y}$ using a product QR algorithm as described next.

1. Compute an economy-size QR decomposition of $\tilde{Y}$ with column pivoting:

   $$\tilde{Y} = Q_1 R_1 \Pi^T, \qquad Q_1 \in \mathbb{R}^{n\times n_\tau(\mathcal{X})},\ R_1 \in \mathbb{R}^{n_\tau(\mathcal{X})\times n_\tau(\mathcal{X})},$$

   where Q_1 has orthonormal columns, R_1 is upper triangular and Π is a permutation.

2. Multiply and permute

   $$\begin{aligned} \tilde{Z} &\leftarrow \tilde{Z}Q_1 &&\in \mathbb{R}^{n_\tau(\mathcal{X})\times n_\tau(\mathcal{X})}, \\ \tilde{Y} &\leftarrow R_1\Pi^T &&\in \mathbb{R}^{n_\tau(\mathcal{X})\times n_\tau(\mathcal{X})}. \end{aligned}$$

3. Compute product Hessenberg form of $\tilde{Z}\tilde{Y}$

   $$H_1 H_2 \leftarrow U_1^T \tilde{Z} U_2 U_2^T \tilde{Y} U_1,$$

   where H_1 is upper Hessenberg and H_2 upper triangular [108, Section 4.2.3].

4. Compute product Schur decomposition

   $$S_1 S_2 \leftarrow W_1^T H_1 W_2 W_2^T H_2 W_1,$$

   where S_1 is in real Schur form and S_2 is upper triangular [108, Section 4.2.1]. The eigenvalues are ordered by descending magnitude.

5. The invariant subspace of $\tilde{Z}\tilde{Y}$ is spanned by the columns of U_1W_1:

   $$\tilde{Z}\tilde{Y}\ U_1W_1 = U_1W_1\ S_1S_2.$$

Algorithm 10 Approximate Cross-Gramian approach for LTI systems (2.1)

INPUT: $A_{\mathcal{H}} \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, tol, τ, ϵ.

OUTPUT: $\hat{A} \in \mathbb{R}^{r\times r}$, $\hat{B} \in \mathbb{R}^{r\times m}$, $\hat{C} \in \mathbb{R}^{p\times r}$, $\hat{D} \in \mathbb{R}^{p\times m}$; reduced order r, error bound δ.

1: Compute low-rank factors $\tilde{Y} \in \mathbb{R}^{n\times n_\tau(\mathcal{X})}$, $\tilde{Z} \in \mathbb{R}^{n_\tau(\mathcal{X})\times n}$ of the cross-Gramian $\mathcal{X}$ by Algorithm 4.

2: Compute right eigenspace U_1W_1 of $\tilde{Z}\tilde{Y}$ by (product) QR algorithm with eigenvalues in decreasing (by magnitude) order $|\lambda_1| \geq \cdots \geq |\lambda_{n_\tau(X)}|$. Adaptive choice of r by tol: $\delta = 2\sum_{j=r+1}^{n_\tau(\mathcal{X})} |\lambda_j| \leq \text{tol}$.

3: Compute right dominant invariant subspace V_r of $\tilde{Y}\tilde{Z}$:

$$V_r = \tilde{Y}(U_1W_1(:, 1:r)).$$

4: Compute left dominant invariant subspace W_l of $\tilde{Y}\tilde{Z}$ by (product) QR algorithm applied to $\tilde{Y}^T\tilde{Z}^T$.

5: Compute QR decompositions $V_r = Q_rR_r$, $W_l = Q_lR_l$ and projection matrices $T_r = Q_r$, $T_l = (Q_l^TQ_r)^{-1}Q_l^T$.

6: Compute reduced-order model:

$$\hat{A} = T_lA_{\mathcal{H}}T_r,\ \hat{B} = T_lB,\ \hat{C} = CT_r,\ \hat{D} = D.$$

The size of the reduced-order system is determined by the ordered eigenvalues of $\tilde{Z}\tilde{Y}$ and by a given error tolerance using the criterion (5.7). Then, the dominant right invariant subspace of the approximate cross-Gramian $\tilde{Y}\tilde{Z}$ corresponding to the r largest (in magnitude) eigenvalues can be derived by the first r columns of U_1W_1, setting $V_r := \tilde{Y}(U_1W_1(:, 1:r)) \in \mathbb{R}^{n\times r}$. The left dominant invariant subspace $W_l \in \mathbb{R}^{n\times r}$ of $\tilde{Y}\tilde{Z}$

$$W_l^T(\tilde{Y}\tilde{Z}) = \Lambda_1W_l^T$$

is computed analogously via a product QR algorithm applied to $\tilde{Y}^T\tilde{Z}^T$. The projection matrices for model order reduction are obtained similar as in the *balancing-free SR method* [165] by an orthogonalization of V_r and W_l. For this purpose we compute two economy-size QR decompositions

$$V_r = Q_rR_r \quad \text{and } W_l = Q_lR_l, \quad Q_r, Q_l \in \mathbb{R}^{n\times r},$$

setting $T_r = Q_r$, $T_l = (Q_l^TQ_r)^{-1}Q_l^T$ and obtain a reduced-order system by projection (5.3). All steps of the cross-Gramian approach are summarized in Algorithm 10.

Remark 5.3.2 (MIMO systems) The cross-Gramian can also be computed for non-symmetric, square (i.e. $m = p$) systems, including some information about the controllability and the observability of the system. Model reduction can be applied in the same way, but without theoretical background and

therefore without any guaranty for the quality of the reduced-order system. In [156] it is proposed to embed a general MIMO system into a symmetric one of the same order but with more inputs and outputs. The order of the resulting symmetric, square system is reduced by a cross-Gramian approach. □

5.3.2 Numerical Results

Besides the data-sparse solver for Sylvester equations in Algorithm 4, all computational steps of the cross-Gramian (CG) approach (Algorithm 10) are computed in dense arithmetic. For the product QR algorithm, we employ the routine MB03VD from the SLICOT Library [27, 151] to compute the product Hessenberg form of a product of matrices without evaluating any part of the product. The matrix product is transformed further to product real Schur canonical form by the HAPACK [94, 108] routine DHGPQR and reordered by DTGSRT such that the magnitudes of the eigenvalues appear in decreasing order.

First, the approximate cross-Gramian (CG) approach is applied to the SISO systems from the previous chapter. We compare the frequency response errors $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ obtained by the cross-Gramian approach with those of the approximate BT method, see the upper plots in Figures 5.12, 5.13 and 5.14. For the error tolerance 10^{-4}, the reduced order computed by Algorithm 10 is equal to the reduced order obtained by the approximate BT method. In all plots we observe that both curves (for the BT errors and the errors obtained by using the cross-Gramian approach) and the corresponding error bounds δ nearly coincide. In the lower plots of the Figures 5.12, 5.13 and 5.14, we also depict the errors between the original and the reduced-order systems obtained by the new method. The errors $\|G - \hat{G}\|_{\mathcal{H}_\infty}$ and $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{H}_\infty}$ are nearly the same. Thus, using the $\mathcal{H}$-matrix format in the computations does not contribute much to the errors between original and reduced-order system.

Next we apply Algorithm 10 to a symmetric MIMO system as obtained by the spatial discretization of the heat equation with $n = 16,384$ grid points and $m = 8$, setting $C = B^T$. The reduced order for tol $= 10^{-4}$ is $r = 11$. In Figure 5.15 the errors plots for several of the 64 input/output channels of the system are depicted. All graphs satisfy the computed error estimate of $\delta = 8.1 \times 10^{-5}$.

Finally, we examine the accuracy of all computed reduced-order systems by the magnitudes for the frequency responses. We compare the magnitudes for the frequency responses of the original system, of $\Sigma(A_{\mathcal{H}}, B, C)$ and of the reduced-order systems obtained by BT, SPA and by the cross-Gramian approach for the examples based on the heat equation and the convection-diffusion equation. In Figures 5.16 and 5.17 it is observed that the graphs of the reduced-order systems produced by approximate BT and by the CG approach are not distinguishable. The reduced-order systems obtained by SPA yield bigger errors for high frequencies. Visible deviations of all reduced-order systems from the unreduced systems $\Sigma(A, B, C)$ and $\Sigma(A_{\mathcal{H}}, B, C)$ only occur when the magnitudes for the frequency responses are very low.

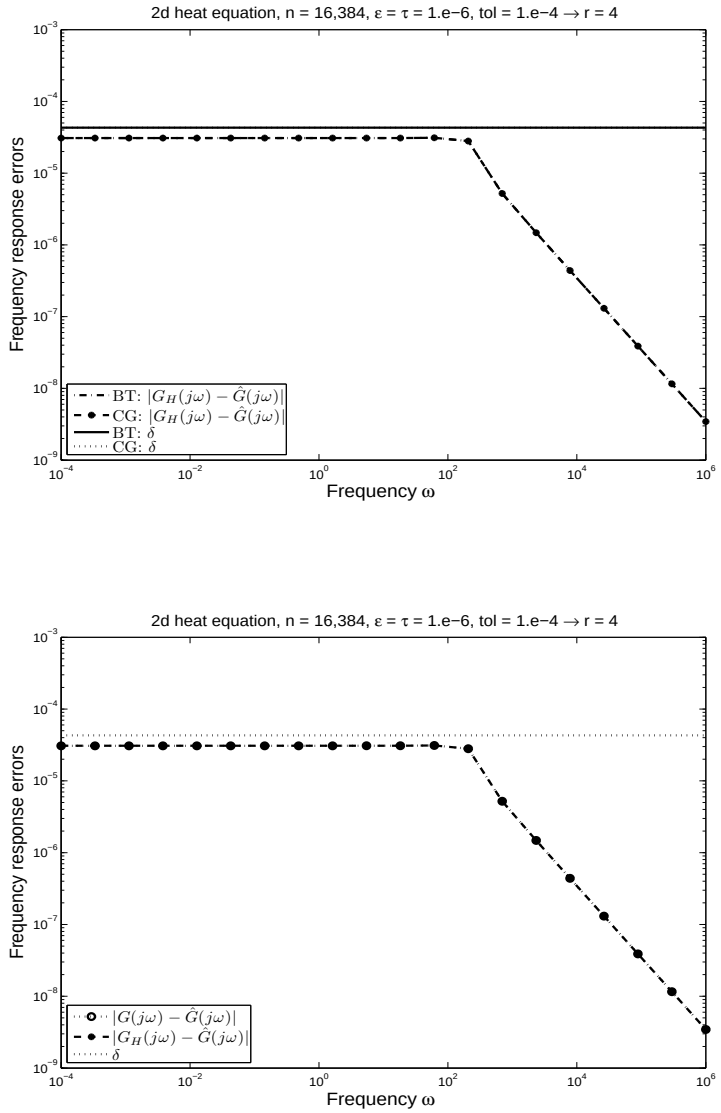


Figure 5.12: Frequency response errors for the two-dimensional heat equation using the cross-Gramian approach as described in Algorithm 10.

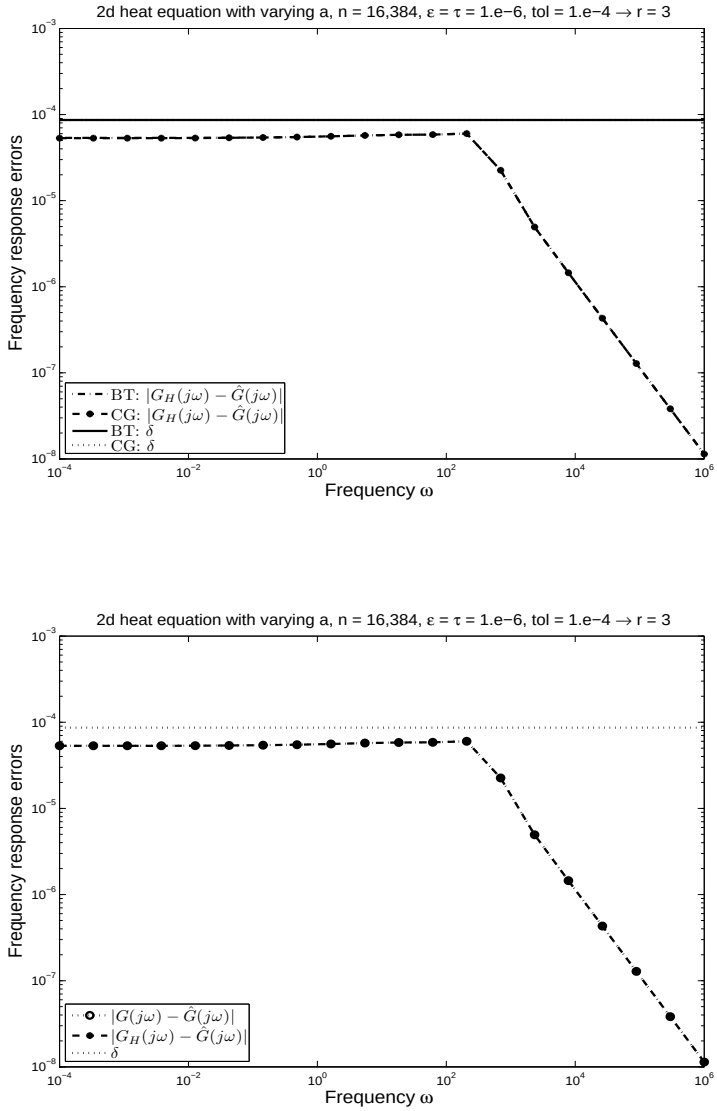


Figure 5.13: Frequency response errors for the two-dimensional heat equation with varying diffusion using the cross-Gramian approach as described in Algorithm 10.

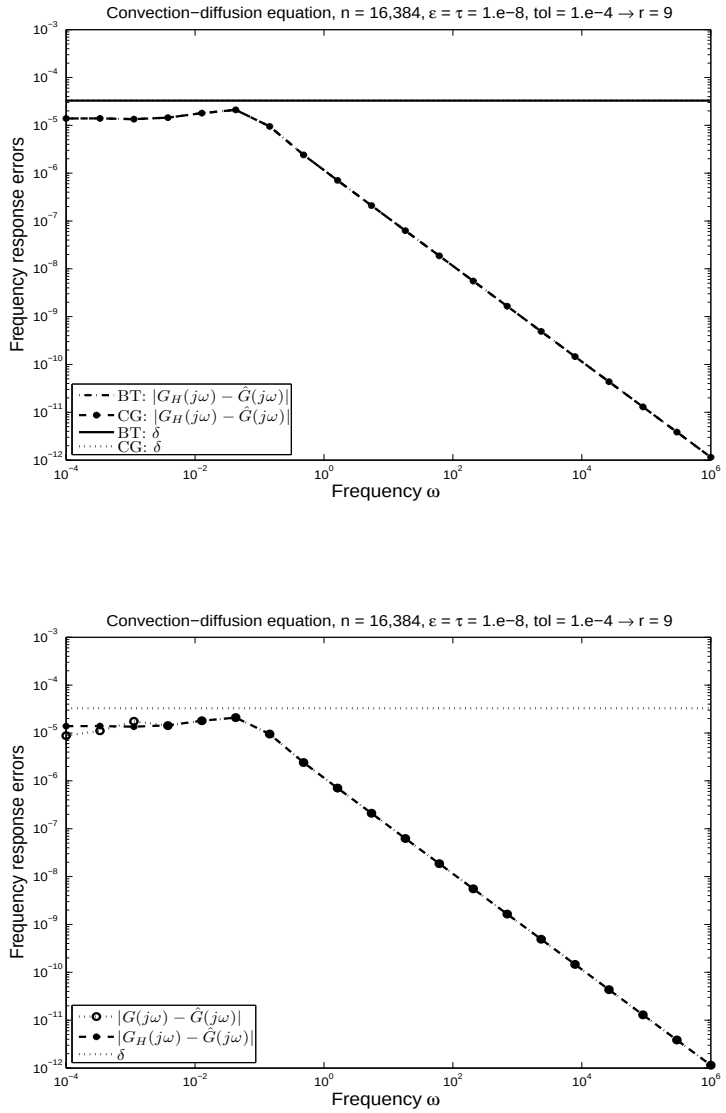


Figure 5.14: Frequency response errors for the convection-diffusion equation using the cross-Gramian approach as described in Algorithm 10.

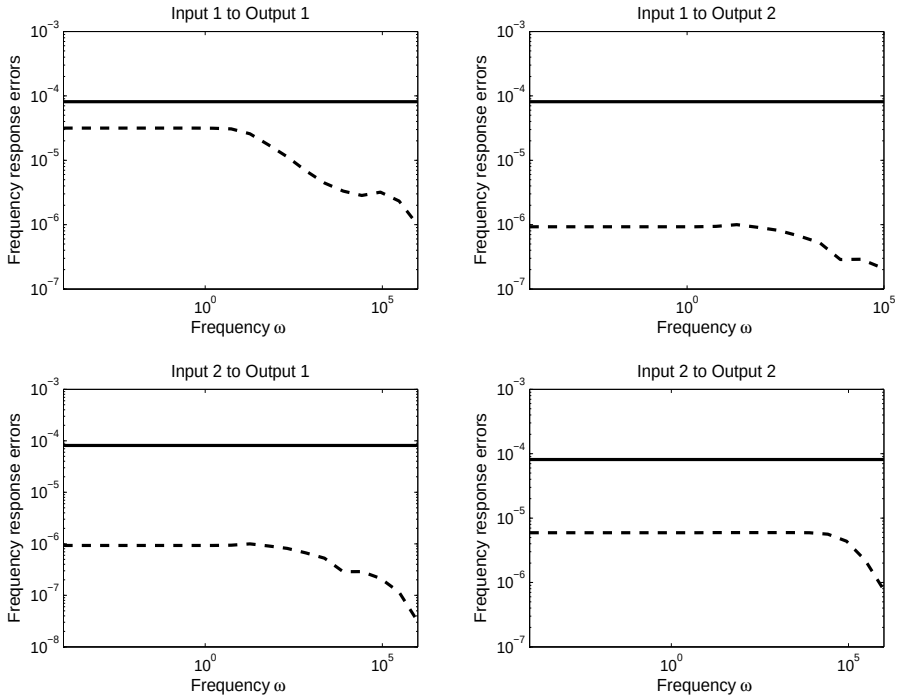


Figure 5.15: Frequency response errors for the two-dimensional heat equation using the cross-Gramian approach as described in Algorithm 10.

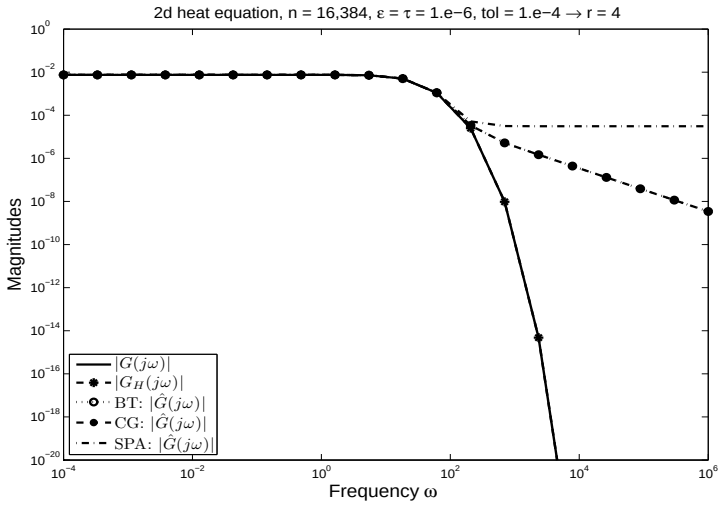


Figure 5.16: Frequency responses of the two-dimensional heat equation for BT, the CG approach and SPA.

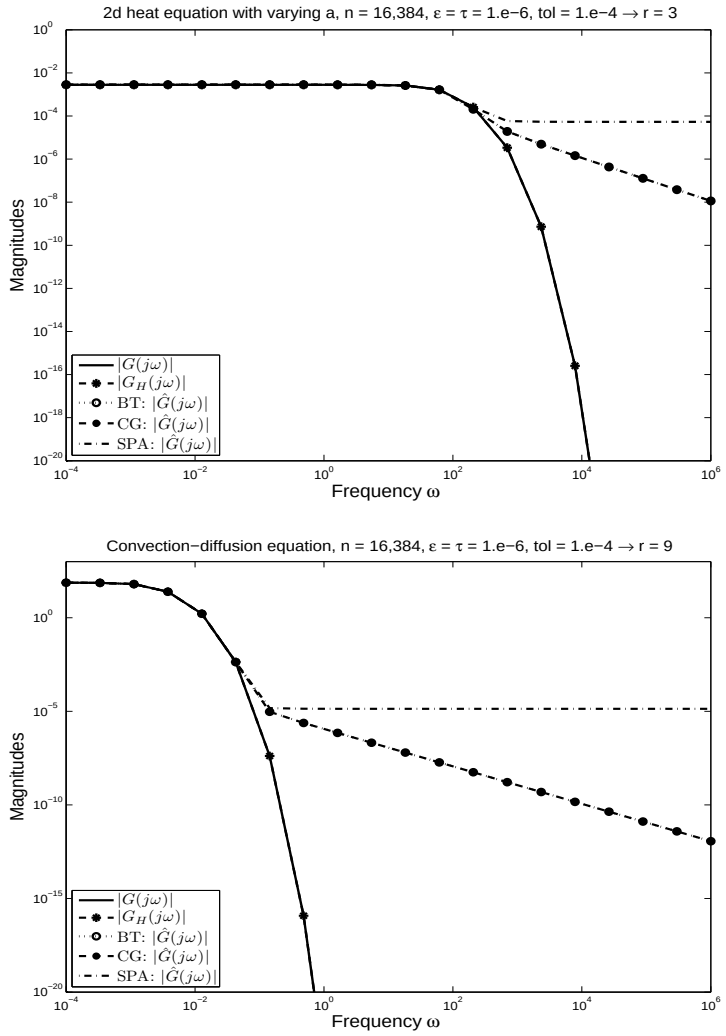


Figure 5.17: Frequency responses for BT, the CG approach and SPA.

Chapter 6

Further Applications

In this chapter we consider two further applications where the data-sparse sign-function solvers can be used to reduce the complexity and storage requirements of the algorithms. Both applications are closely related to model reduction for large-scale systems. In Section 6.1 we consider a modification of the usual balanced truncation method for large-scale, continuous-time systems where in contrast to the methods in Chapter 5 the matrix A is allowed to have unstable eigenvalues. In Section 6.2 an approach for solving an optimal control problem with inequality constraints for the control is presented. Model reduction of the underlying discrete-time LTI system makes the application of standard optimization software for the solution of a quadratic programming problem possible.

6.1 Model Reduction for Unstable Systems

As opposed to the previous sections, we now allow for unstable systems (2.1) satisfying

$$\Lambda(A) \cap \mathbb{C}^+ \neq \emptyset, \quad \Lambda(A) \cap \jmath\mathbb{R} = \emptyset. \tag{6.1}$$

Under these assumptions, the time domain representation of the Gramians (2.15) is not defined. However, there is an equivalent definition of the Gramians in frequency domain and this carries over to unstable systems satisfying (6.1) [155, 167]:

$$\mathcal{P} = \frac{1}{2\pi} \int_{-\infty}^{\infty} (\jmath\omega I_n - A)^{-1} B B^T (-\jmath\omega I_n - A^T)^{-1} d\omega, \tag{6.2}$$

$$\mathcal{Q} = \frac{1}{2\pi} \int_{-\infty}^{\infty} (-\jmath\omega I_n - A^T)^{-1} C^T C (\jmath\omega I_n - A)^{-1} d\omega. \tag{6.3}$$

The following result, stated in [167], describes the numerical solution of the Gramians in frequency domain by the solution of matrix equations.

Theorem 6.1.1 [167, Theorem 2] *Suppose (A, B) is stabilizable, (A, C) is detectable, $\Lambda(A) \cap \jmath\mathbb{R} = \emptyset$ and let X and Y denote the unique stabilizing solutions*

(i.e. $\Lambda(A-BB^TX)\subset\mathbb{C}^-$, $\Lambda(A-YC^TC)\subset\mathbb{C}^-$) of the two algebraic Bernoulli equations

$$\begin{aligned} A^TX+XA-XBB^TX &= 0, && (6.4)\\ AY+YA^T-YC^TCY &= 0, && (6.5)\end{aligned}$$

respectively. Then, the Gramians $\mathcal{P}$ and $\mathcal{Q}$ in (6.2), (6.3) are given by the solutions of two Lyapunov equations

$$\begin{array}{rcrcrcl} (A-BB^TX)\,\mathcal{P} & + & \mathcal{P}(A-BB^TX)^T & + & BB^T & = & 0,\\ (A-YC^TC)^T\mathcal{Q} & + & \mathcal{Q}(A-YC^TC) & + & C^TC & = & 0. \end{array} \qquad (6.6)$$

It is shown in [167] that a balanced realization and a reduced-order system based on these Gramians can be computed in the same way as for stable systems by the usual balanced truncation method. Additionally, the balanced truncation error bound (5.4) holds for the $\mathcal{L}_\infty$-norm of the error between original and truncated system [167, Theorem 4]. In the following we investigate the numerical solution of ABEs, exemplarily for computing the stabilizing solution X of (6.4). Very useful for an efficient implementation is the following observation.

Theorem 6.1.2 [24] *If (A,B) is stabilizable, (6.1) and X is the unique stabilizing solution of (6.4), then*

$$\operatorname{rank}(X)=\ell,$$

where ℓ is the number of eigenvalues of A in $\mathbb{C}^+$.

It is well known that a solution X of an ABE (which is a homogeneous ARE) can be derived from the invariant subspace of the associated Hamiltonian matrix Z as

$$\underbrace{\begin{bmatrix} A & BB^T\\ 0 & -A^T\end{bmatrix}}_{=:Z}\begin{bmatrix} I_n\\ -X\end{bmatrix}=\begin{bmatrix} I_n\\ -X\end{bmatrix}(A-BB^TX),$$

see for instance [112]. Thus, if (A,B) is stabilizable and (6.1) holds, the unique stabilizing solution X of (6.4) can be computed by the Z-invariant subspace corresponding to the stable eigenvalues, i.e.

$$\Lambda(A-BB^TX)\subset(\Lambda(Z)\cap\mathbb{C}^-).$$

Using spectral projection, the kernel of the projector onto the anti-stable Z-invariant subspace $P_+:=(I_{2n}+\operatorname{sign}(Z))/2$ (see Lemma 3.1.2 d)) describes the stable Z-invariant subspace. Thus, the stabilizing solution X can be derived from the linear least-squares problem

$$\frac{1}{2}(I_{2n}+\operatorname{sign}(Z))\begin{bmatrix} I_n\\ -X\end{bmatrix}=0. \qquad (6.7)$$

To summarize, for computing the stabilizing solution X of (6.4), the matrix sign function of the associated Hamiltonian Z has to be computed followed by solving (6.7).

6.1.1 The Data-Sparse Approach

In order to apply model order reduction to large-scale systems satisfying (6.1), the hierarchical matrix format is incorporated in the computations as described in the following. The sign function of the Hamiltonian

$$Z = \begin{bmatrix} A & BB^T \\ 0 & -A^T \end{bmatrix}$$

is computed analogously as in Chapter 4 for the numerical solution of Lyapunov equations. By the block structure of Z, the iteration splits into two parts and again, we approximate the expensive part for the A-iterates in the hierarchical matrix format. The only difference to the sign function solver in Section 4.2 is the stopping criterion which has to be modified to

$$\|A_{j+1} - A_j\|_2 \leq \mathrm{tol}\, \|A_{j+1}\|_2.$$

It is observed that in usual applications which stem from the semi-discretization of some elliptic partial differential operator the B-iterates and thus the solution X have a small (numerical) rank. Note that by simple transformations of (6.4) we can derive properties of the solution X of the Bernoulli equation (6.4) from the stable Lyapunov equation

$$(A - BB^T\tilde{X})^T X + X(A - BB^T\tilde{X})A + \tilde{X}BB^T\tilde{X} = 0, \tag{6.8}$$

where $\tilde{X}$ denotes the stabilizing solution of (6.4). Since (6.8) is equivalent to (6.4), $\tilde{X}$ is the unique solution of (6.8). The constant term in the Lyapunov equation is of low rank and thus the solution X is expected to be of low rank too, at least numerically. Therefore, the expected low numerical rank of the Bernoulli solution is revealed by row compression using an RRLQ factorization. When the sign function iteration is converged, using the notations

$$A_\infty := \lim_{j\to\infty} A_j, \quad B_\infty := \lim_{j\to\infty} B_j,$$

for the limits of the sign function iteration and s for the numerical rank of $B_\infty \in \mathbb{R}^{n\times s}$, we obtain

$$\mathrm{sign}\,(Z) = \begin{bmatrix} A_\infty & B_\infty B_\infty^T \\ 0 & -A_\infty^T \end{bmatrix}.$$

Including this expression for the sign of Z in (6.7), the linear least-squares problem (6.7) is equivalently given by

$$\underbrace{\begin{bmatrix} B_\infty B_\infty^T \\ I_n - A_\infty^T \end{bmatrix}}_{\tilde{A}} X = \underbrace{\begin{bmatrix} I_n + A_\infty \\ 0_n \end{bmatrix}}_{\tilde{b}}.$$

It admits a unique solution if $\mathrm{rank}(\tilde{A}) = n$. Since $\mathrm{rank}(I_n - A_\infty^T) = n - \ell$, we assume that $\mathrm{rank}(B_\infty B_\infty^T) \geq \ell$. To proceed the computations in low complexity we solve the problem using the normal equations

$$\tilde{A}^T\tilde{A}\,X = \tilde{A}^T\tilde{b},$$

thereby exploiting that B_∞ is of low rank and A_∞ is stored as $\mathcal{H}$-matrix. The matrices involved are computed in the following way:

$$\begin{aligned} \underbrace{\tilde{A}^T\tilde{A}}_{\mathcal{H}-matrix} &= \underbrace{B_\infty B_\infty^T B_\infty B_\infty^T}_{low\ rank} + \underbrace{(I_n - A_\infty)}_{\mathcal{H}-matrix}\underbrace{(I_n - A_\infty)^T}_{\mathcal{H}-matrix} \\ \underbrace{\tilde{A}^T\tilde{b}}_{low\ rank} &= \underbrace{B_\infty\,[B_\infty^T + B_\infty^T A_\infty]}_{low\ rank} \end{aligned}$$

where the notation "low rank" indicates that the matrix is stored as product of two rectangular matrices. Using the $\mathcal{H}$-Cholesky-decomposition of $\tilde{A}^T\tilde{A}$ and $\mathcal{H}$-based forward and backward substitutions, we compute the stabilizing solution X of (6.4) as low rank matrix, i.e., $X = X_1 X_2^T,\ X_1, X_2 \in \mathbb{R}^{n\times s}$, see Algorithm 11.

Algorithm 11 Calculate low-rank factors $X_1, X_2 \in \mathbb{R}^{n\times s}$ of stabilizing solution $X = X_1 X_2^T$ for $A^T X + XA - XBB^T X = 0$.

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, tol, ϵ, τ

OUTPUT: Approximate low-rank factors $X_1, X_2 \in \mathbb{R}^{n\times s}$ of the solution X.

1: $A_0 \leftarrow (A)_\mathcal{H}$
2: $B_0 \leftarrow B$
3: $j = 0$
4: **while** $\|A_{j+1} - A_j\|_2 \le \text{tol}\,\|A_{j+1}\|_2$ **do**
5: Compute the approximate inverse $A_{\mathcal{H},j}^{-1}$ by Algorithm 3.
6: $A_{j+1} \leftarrow \frac{1}{2}(A_j \oplus A_{\mathcal{H},j}^{-1})$
7: $B_{j+1} \leftarrow \frac{1}{\sqrt{2}}\begin{bmatrix} B_j, & A_{\mathcal{H},j}^{-1}B_j \end{bmatrix}$
8: Compute an RRLQ factorization

$$B_{j+1} = \Pi L\, Q = \Pi \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} Q$$

with $\|L_{22}\|_2 < \tau\|B_{j+1}\|_2$ and $L_{11} \in \mathbb{R}^{s\times s}$.
9: Compress columns of B_{j+1} to size s:

$$B_{j+1} \leftarrow \Pi \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}.$$

10: $j = j + 1$
11: **end while**
12: $(\tilde{A}^T\tilde{A})_\mathcal{H} \leftarrow B_j B_j^T B_j B_j^T + (I_n \ominus A_j) \odot (I_n \ominus A_j)^T$
13: Compute $\mathcal{H}$-Cholesky-decomposition of $(\tilde{A}^T\tilde{A})_\mathcal{H}$: $L_\mathcal{H} L_\mathcal{H}^T \leftarrow (\tilde{A}^T\tilde{A})_\mathcal{H}$.
14: Compute Y by $\mathcal{H}$-based forward substitution: $L_\mathcal{H} Y = B_j$.
15: Compute X_1 by $\mathcal{H}$-based backward substitution: $L_\mathcal{H}^T X_1 = Y$.
16: $X_2 \leftarrow B_j^T + B_j^T A_j$.

To reduce the dimension of an unstable LTI system with (6.1) we apply Algorithm 11 for the solution of the Bernoulli equations (6.4) and (6.5). Thus, we obtain low-rank solution factors of X and Y:

$$X = X_1 X_2^T, \qquad Y = Y_1 Y_2^T.$$

Following Theorem 6.1.1, the system Gramians $\mathcal{P}$ and $\mathcal{Q}$ are obtained by solving the two Lyapunov equations (6.6) where the two large-scale matrices $(A - BB^T X)$ and $(A - YC^T C)$ are approximated as $\mathcal{H}$-matrices. This can be done in an efficient way since A is approximated as $\mathcal{H}$-matrix, B has only a small number of columns and C a small number of rows. Furthermore, the matrices X and Y are computed in low-rank factored form by the proposed data-sparse solver for ABEs. Thus, the matrices are computed as the sums of $\mathcal{H}$-matrices and an Rk-matrices

$$A_{\mathcal{H}} - B(X_2 X_1^T B)^T, \quad A_{\mathcal{H}} - (Y_1 Y_2^T C^T)C,$$

which can be approximated in the set of $\mathcal{H}$-matrices.

Remark 6.1.3 Note that it is also possible to compute the solution of an ABE directly in $\mathcal{H}$-matrix format. Since an ABE is a homogeneous ARE, the methods proposed in [79] for solving algebraic matrix Riccati equations can be used. □

In contrast to the usual BT implementations for stable systems, the Lyapunov equations (6.6) have to be solved separately since they do not have the same coefficient matrix. Once the approximate Gramians $\mathcal{P}$ and $\mathcal{Q}$ in (6.2) and (6.3), repectively, are computed, we proceed with computing the projection matrices as in the approximate BT method in Algorithm 8, see Section 5.1.1.

6.1.2 Numerical Results

In this section we examine the accuracy and complexity of Algorithm 11 for the numerical solution of Bernoulli equations. Using this solver as first computational step in a model order reduction approach for unstable systems with (6.1) as described in Section 6.1.1, we examine the errors between the original system and the computed reduced-order one.

As exemplary system we consider the following reaction-diffusion equation

$$\frac{\partial \mathbf{x}}{\partial t}(t,\xi) = \Delta \mathbf{x}(t,\xi) + c\mathbf{x}(t,\xi) + b(\xi)u(t), \qquad \xi \in \Omega,\, t \in (0,\infty),$$

which is discretized in space by finite elements as introduced in Example 4.1.13, leading to the LTI system

$$\dot{x}(t) = \underbrace{(\tilde{A} + cI_n)}_{:=A} x(t) + Bu(t). \tag{6.9}$$

For the problem sizes $n = 4096$ and $n = 16,384$, we choose the parameter c such that one eigenvalue of the coefficient matrix A has positive real part: $\lambda \approx 0.25$. We compute the relative residual

$$\frac{\|A^T X + XA - XBB^T X\|_F}{2(\|A\|_F \|X\|_F) + \|X\|_F^2 \|BB^T\|_F}$$

of the ABE (6.4) obtained by applying the data-sparse solver 11 to the unstable system. We vary the parameters τ and ϵ and depict the numerical rank of B_∞, the computational time and the accuracy in Table 6.1. We observe a similar dependency of the relative residuals on ϵ as in Table 4.5 where the solutions of the data-sparse Lyapunov solver applied on the pure diffusion problem are shown. The method needs more iteration steps to fulfill the stopping criterion and, consequently, the numerical ranks are increased compared to those of the stable problem in Section 4.2.2. Nevertheless, accurate solutions of large-scale ABEs are computed in acceptable time.

n	ϵ	τ	# it.	$n_\tau(B_\infty)$	time [sec]	rel. residual
4096	1.e-04	1.e-04	26	36	261	7.1e-06
	1.e-06	1.e-04	22	14	391	1.8e-07
	1.e-08	1.e-04	19	14	635	3.8e-08
	1.e-06	1.e-06	22	31	395	1.8e-07
	1.e-08	1.e-06	19	21	636	3.7e-08
	1.e-08	1.e-08	19	39	639	3.7e-08
16,384	1.e-04	1.e-04	27	34	2376	2.3e-05
	1.e-06	1.e-04	26	15	4235	5.2e-07
	1.e-08	1.e-04	22	14	7136	6.1e-09
	1.e-06	1.e-06	26	42	4273	5.2e-07
	1.e-08	1.e-06	22	23	7150	6.0e-09
	1.e-08	1.e-08	22	42	7183	5.9e-09

Table 6.1: Accuracy and rank $n_\tau(B_\infty)$ of the computed ABE solution for different problem sizes and parameter combinations.

Based on the solver for ABEs we reduce the dimension of the original unstable system (6.9) of order $n = 4096$. We compute a reduced-order system of order $r = 6$ for the reaction-diffusion equation using the fixed parameter choice of $\epsilon = \tau = 10^{-6}$. The eigenvalues of A and of $\hat{A}$ closest to the imaginary axis are depicted in Figure 6.1. It is observed that the unstable eigenvalue of the original system is preserved in the reduced-order system. In Figure 6.2 it is shown that the frequency response errors $\|G_{\mathcal{H}} - \hat{G}\|_{\mathcal{L}_\infty}$ satisfy the error estimate $\delta = 2.36 \times 10^{-5}$.

6.2 Linear Quadratic Optimal Control

In the following we describe one possible application for model order reduction of discrete-time systems. The problem results from the full discretization of a control problem which includes inequality constraints for the control of the underlying PDE. The discrete optimal control problem is converted into a nonlinear programming problem of very large dimension. In order to apply standard optimization software like `quadprog` from the MATLAB Optimization Toolbox, we can reduce the space dimension of the discretized parabolic problem using the approximate BT method from Section 5.1.1.

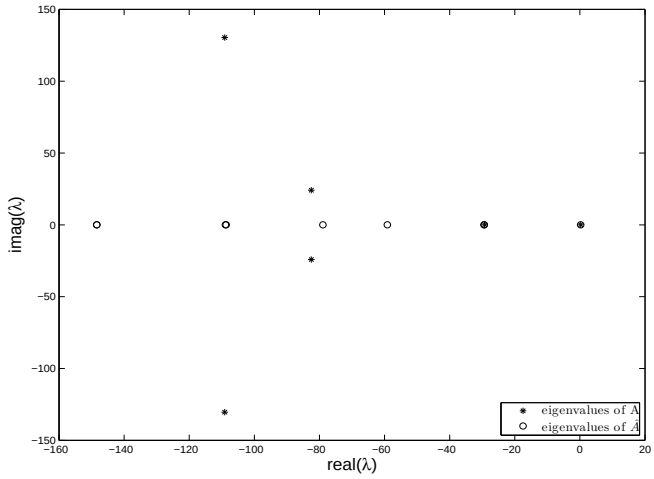


Figure 6.1: Eigenvalues of A and $\hat{A}$ of the unstable LTI system (6.9).

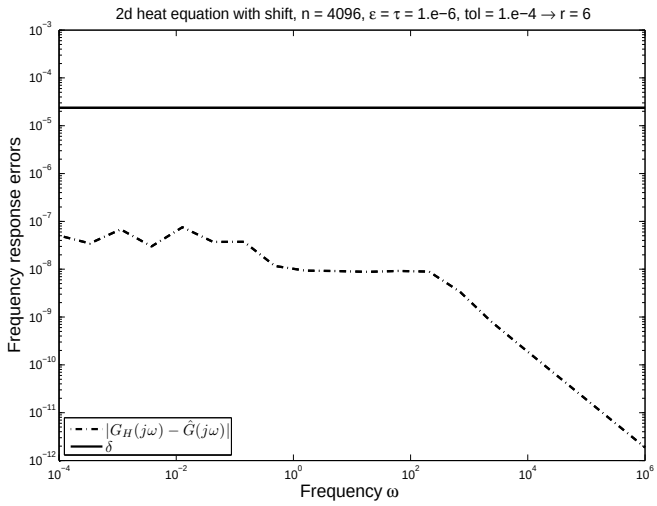


Figure 6.2: Frequency response errors for the unstable LTI system (6.9).

6.2.1 Model Reduction for Linear Parabolic Optimal Control Problems

In this section we introduce an approach for the numerical solution of linear-quadratic optimal control problems with inequality control constraints. For the theoretical background of the optimal control problem we refer to [162, Section 3.5]. For the existence and uniqueness of the solution for a discrete optimal control problem, rewritten as nonlinear programming problem, see [59].

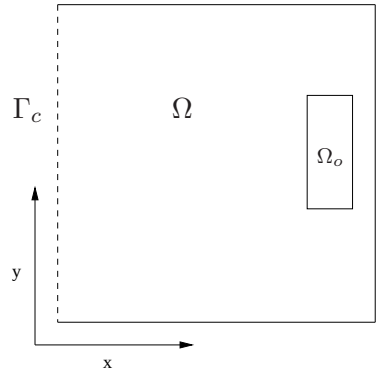


Figure 6.3: Domain description for the two-dimensional heat equation.

We investigate the numerical solution of the following continuous optimal control problem for heat transfer in the domain $\Omega = [0,1]^2$ shown in Figure 6.3:

$$\min J(\mathbf{y},\mathbf{u}) := \frac{1}{2}\int_0^T\int_{\Omega_o}(\mathbf{y}(t,\xi)-y_\Omega(\xi))^2 d\xi dt + \frac{\lambda}{2}\int_0^T\int_{\Gamma_c}\mathbf{u}(t,\xi)^2 d\xi\, dt, \quad (6.10)$$

subject to

$$\begin{array}{lll} \frac{\partial \mathbf{x}}{\partial t}(t,\xi) - \Delta \mathbf{x}(t,\xi) & = 0, & \xi\in\Omega,\ t\in(0,T),\\ \mathbf{x}(0,\xi) & = \mathbf{x}_0(\xi), & \xi\in\Omega,\\ \frac{\partial \mathbf{x}}{\partial \nu}(t,\xi) & = \alpha(\mathbf{x}(t,\xi)-\mathbf{u}(t,\xi)), & \xi\in\Gamma_c,\ t\in(0,T),\\ \frac{\partial \mathbf{x}}{\partial \nu}(t,\xi) & = 0, & \xi\in\Gamma\setminus\Gamma_c,\ t\in(0,T),\\ \mathbf{y}(t,\xi) & = \mathbf{x}(t,\xi)_{|\Omega_o}, & \xi\in\Omega_o,\ t\in(0,T). \end{array} \quad (6.11)$$

In this setting, $\lambda > 0$ is a regularization parameter for the optimal control problem, $\mathbf{x}_0$ is the initial temperature, $T > 0$ a fixed control horizon, $\alpha \in \mathbb{R}$ describes the heat transfer and $\mathbf{u}(\cdot)$ is the control applied on one part of the boundary, see Figure 6.3. The control objective is to control the temperature $\mathbf{x}(\cdot)$ to a prescribed temperature distribution $y_\Omega(\cdot)$ in Ω_o using the tracking type functional (6.10). For the control $\mathbf{u}(\cdot)$ the following (pointwise) inequality constraints are imposed:

$$u_a(t,\xi) \le u(t,\xi) \le u_b(t,\xi), \quad \xi\in\Gamma_c,\ t\in(0,T).$$

We follow the computational strategy of "*Discretize Then Optimize*" [39], the continuous problem is transformed into a discrete quadratic control problem by discretizing (6.10) and (6.11) in space and time. For the space discretization we use a uniform triangulation of the spatial domain Ω with n grid points and n linear ansatz functions $\varphi_1,\ldots,\varphi_n$, similar as in Example 4.1.13. A subset of p grid points is located in the subdomain Ω_o, the corresponding ansatz functions are noted by $\varphi_1,\ldots,\varphi_p$. The boundary Γ_c is discretized with m points, using the ansatz space $\psi_1,\ldots,\psi_m$. We approximate the states and the controls by

$$\mathbf{x}(t,\xi) \approx \sum_{j=1}^{n} x_j(t)\varphi_j(\xi), \quad \mathbf{y}(t,\xi) \approx \sum_{j=1}^{p} y_j(t)\varphi_j(\xi), \quad \mathbf{u}(t,\xi) \approx \sum_{j=1}^{m} u_j(t)\psi_j(\xi).$$

The FEM stiffness matrix $A \in \mathbb{R}^{n\times n}$ and the mass matrix $E \in \mathbb{R}^{n\times n}$ arising in the space discretization of (6.11) are computed as described in (2.31). The entries of the matrices $B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{n\times p}$ are given as in Example 4.1.13. Additionally, the mass matrices $E_o \in \mathbb{R}^{p\times p}$ and $E_c \in \mathbb{R}^{m\times m}$ are computed on Ω_o and Γ_c, respectively, using the corresponding ansatz spaces. With

$$x(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{pmatrix},\ y(t) = \begin{pmatrix} y_1(t) \\ \vdots \\ y_p(t) \end{pmatrix},\ y_\Omega = \begin{pmatrix} y_{\Omega,1} \\ \vdots \\ y_{\Omega,p} \end{pmatrix},\ u(t) = \begin{pmatrix} u_1(t) \\ \vdots \\ u_m(t) \end{pmatrix},$$

the semi-discretization leads to the following intermediate control problem:

$$\min \tilde{J}(x,u) :=$$

$$\frac{1}{2}\int_0^T \left(x(t)^T C^T E_o C x(t) - 2y_\Omega^T E_o C x(t) + y_\Omega^T E_o y_\Omega\right) dt + \frac{\lambda}{2}\int_0^T u(t)^T E_c u(t) dt,$$

subject to

$$\begin{array}{rcl} E\dot{x}(t) &=& Ax(t) + Bu(t), \quad x(0) = x_0, \\ y(t) &=& Cx(t), \end{array} \tag{6.12}$$

and a set of constraints

$$u_a(t) \le u(t) \le u_b(t) \quad \text{for } t \in (0,T).$$

We use a simple quadrature rule for the time discretization of the cost functional $\tilde{J}$ and ℓ time steps of equidistant size T_s. The solution x and the control u at the ith time step are denoted by $x_i := x(iT_s)$, $u_i := u(iT_s)$, respectively, for $i = 0, \ldots, \ell$. Using the backward Euler scheme for the time discretization of the LTI system as in Section 4.4.2, we obtain the following discrete control problem:

$$\min \tilde{J}(x_0, \ldots, x_{\ell-1}) = \frac{1}{2}\sum_{k=1}^{\ell-1}(x_k^T T_s C^T E_o C x_k - 2y_\Omega^T T_s E_o C x_k) + \frac{\lambda}{2}\sum_{k=1}^{\ell-1}(u_k^T E_c u_k) \tag{6.13}$$

subject to

$$\begin{array}{rcll} x_{k+1} &=& Ax_k + Bu_k, & x_0 = x^0, \\ y_k &=& Cx_k, & \end{array} \tag{6.14}$$

and

$$u_{a,\,k} \le u_k \le u_{b,\,k}, \quad \text{for } 0 \le k < \ell - 1.$$

By a slight abuse of notation, we denote the matrices in the discrete-time LTI system (6.14) as those in the continuous-time setting although they are computed in a different way, see Section 4.4.2. We want to solve this finite dimensional problem with existing control software. For this purpose, the discrete problem (6.13) to (6.14) is converted into a quadratic programming problem. Based on the vector of unknowns

$$z := (x_0^T, \ldots, x_{\ell-1}^T,\, u_0^T, \ldots, u_{\ell-1}^T)^T \in \mathbb{R}^{\ell n + \ell m},$$

the problem (6.13) to (6.14) is equivalent to

$$\min_{Rz=c} \frac{1}{2} z^T H z + f^T z, \quad \alpha \le z \le \beta, \tag{6.15}$$

with a constraint matrix

$$R := \left(\begin{array}{ccccc|cccc} I & 0 & \dots & \dots & 0 & 0 & \dots & \dots & 0 \\ -A & I & & & & -B & & & \vdots \\ 0 & -A & I & \ddots & \vdots & 0 & -B & & \\ \vdots & \ddots & \ddots & \ddots & 0 & & & \ddots & 0 \\ 0 & \dots & 0 & -A & I & 0 & \dots & 0 & -B \; 0 \end{array}\right) \in \mathbb{R}^{\ell n \times (\ell n + \ell m)}$$

and a matrix for the quadratic terms in the objective

$$H := \begin{pmatrix} T_s C^T E_o C & & & & & & \\ & \ddots & & & & & \\ & & T_s C^T E_o C & & & & \\ & & & \lambda T_s E_c & & & \\ & & & & \ddots & & \\ & & & & & \lambda T_s E_c \end{pmatrix} \in \mathbb{R}^{(\ell n + \ell m) \times (\ell n + \ell m)}$$

and vectors for the initial conditions, for the linear part of the objective and for the control constraints

$$c = \begin{pmatrix} x^0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \; f = \begin{pmatrix} y_\Omega^T T_s E_o C \\ \vdots \\ y_\Omega^T T_s E_o C \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \; \alpha = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ u_{a,\,0} \\ \vdots \\ u_{a,\,\ell-1} \end{pmatrix}, \; \beta = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ u_{b,\,0} \\ \vdots \\ u_{b,\,\ell-1} \end{pmatrix}.$$

The matrices R and H are sparse and very high dimensional. Since this very large dimension limits the applicability of optimization software we reduce the probably large state space dimension n of the discrete system (6.14) before the problem is rewritten into a quadratic programming problem.

We apply the approximate BT method to the discrete-time LTI system (6.14) and compute a system of reduced order r. Using this reduced-order system $\Sigma(\hat{A}, \hat{B}, \hat{C})$ in the subsequent transcription of the discrete problem into the quadratic programming problem reduces the dimensions of the unknown vector z from $\mathbb{R}^{\ell n + \ell m}$ to $\mathbb{R}^{\ell r + \ell m}$. Since all blocks in the large matrices R and H are of reduced size as well, we obtain a constraint matrix of reduced dimension, $\hat{R} \in \mathbb{R}^{\ell r \times (\ell r + \ell m)}$, and a much smaller matrix $\hat{H} \in \mathbb{R}^{(\ell r + \ell m) \times (\ell r + \ell m)}$ corresponding to the quadratic terms in (6.13).

6.2.2 Numerical Results

For the continuous optimal control problem (6.10), (6.11), the following constants are set: for the heat transfer $\alpha = 8.0$, as regularization parameter

$\lambda = 10^{-4}$ and for the control constraints $u_{a,k} = 0$, $u_{b,k} = 50$ for $0 \le k < \ell$. We choose the initial temperature $x_0(\cdot) \equiv 0$ on Ω and y_Ω as plotted in Figure 6.4(a). We discretize the parabolic system (6.11) in space by a uniform triangulation with $16,900$ grid points. 130 grid points are located on the boundary Γ_c and 12 points in the subdomain Ω_o. The time interval $[0, 1]$ is discretized with constant step size $T_s = 0.1$. We use the backward Euler method for the time discretization of the continuous-time LTI system (6.12) and obtain a discrete-time system (6.14) with $n = 16,900$, $m = 130$ and $p = 12$.

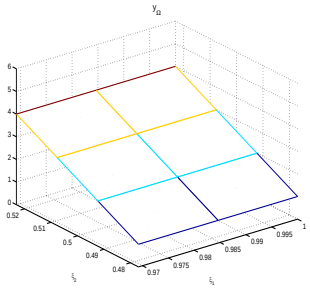
Without model order reduction, the problem dimensions in (6.15) would be the following:

$$R \in \mathbb{R}^{169,000\times 170,300}, \quad H \in \mathbb{R}^{170,300\times 170,300}.$$

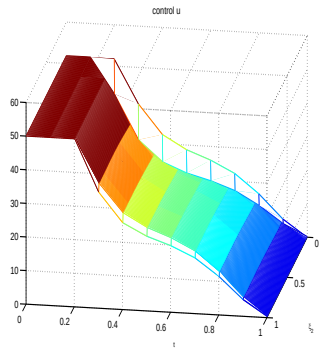
We apply the approximate BT method to (6.14) using a tolerance of 10^{-6} and compute a reduced-order system of order $r = 8$ by an error estimate of $\delta = 9.48 \times 10^{-7}$. Based on the reduced-order system, the discrete control problem (6.13) is formulated as a quadratic programming problem (6.15). Note that the dimensions of the matrices involved are reduced significantly using the reduced-order system with state space dimension $r = 8$ instead of $n = 16,900$

$$\hat{R} \in \mathbb{R}^{80\times 1380}, \quad \hat{H} \in \mathbb{R}^{1380\times 1380}.$$

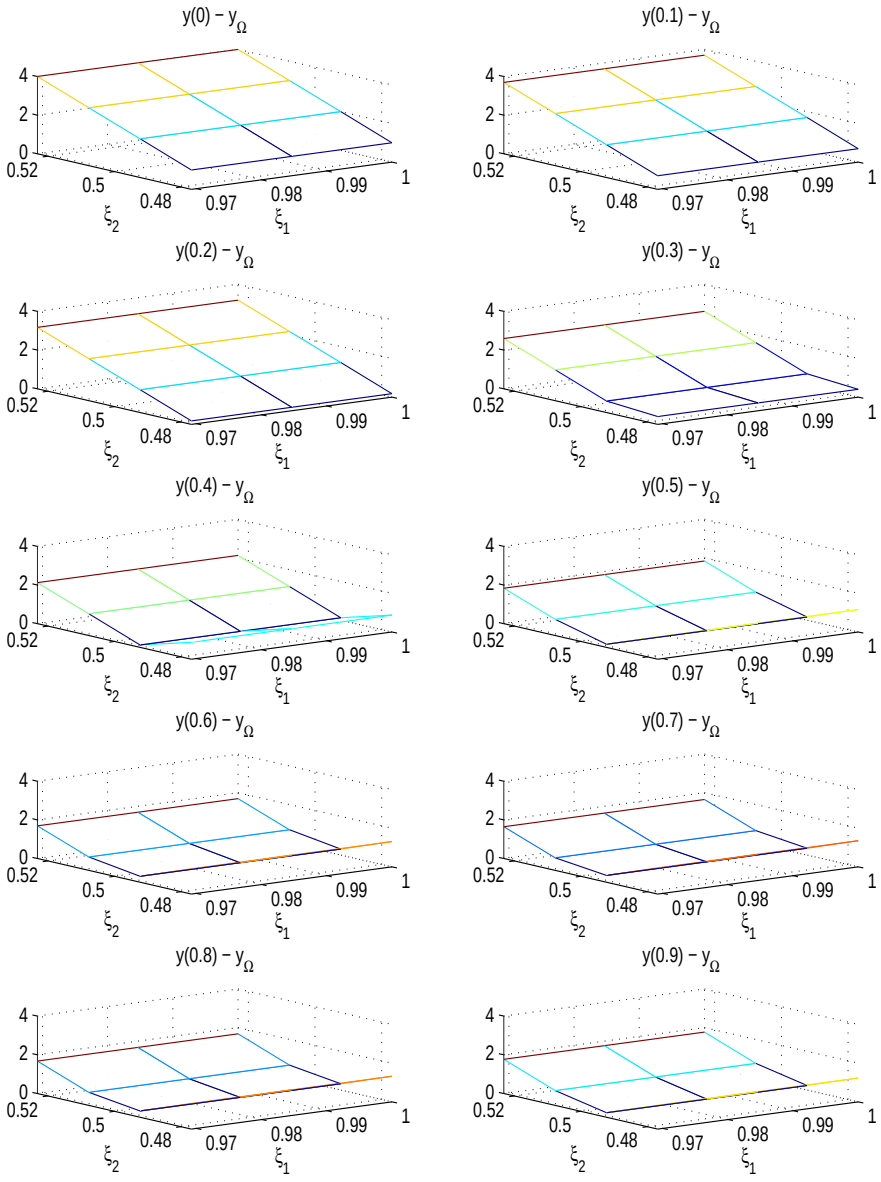
We use the `quadprog` solver from MOSEK ApS [128] which can handle sparse matrices in contrast to `quadprog` from the MATLAB Optimization Toolbox. The solution is computed very quickly, the elapsed time is about 0.095 seconds. The computed optimal control is shown in Figure 6.4(b). It is observed that the control constraints are active at the beginning of the control process. The differences between the controlled state $\mathbf{y}(t, \xi)$ and $y_\Omega(\xi)$ in Ω_o at $t = 0.1, 0.2, \cdots, 0.9$ are plotted in Figure 6.4(c). In Figure 6.5, the snapshots of the optimal solution $\mathbf{x}(t, \xi)$ at $t = 0.1, 0.2, \cdots, 0.9$ are depicted.



(a) $y_\Omega(\xi)$ in Ω_o.



(b) $\mathbf{u}(t,\xi)$ on $\Gamma_c \times (0,T)$.



(c) Differences between $\mathbf{y}(t,\xi)$ and $y_\Omega(\xi)$ in Ω_o

Figure 6.4: Settings and results of the optimal control problem in Section 6.2.2.

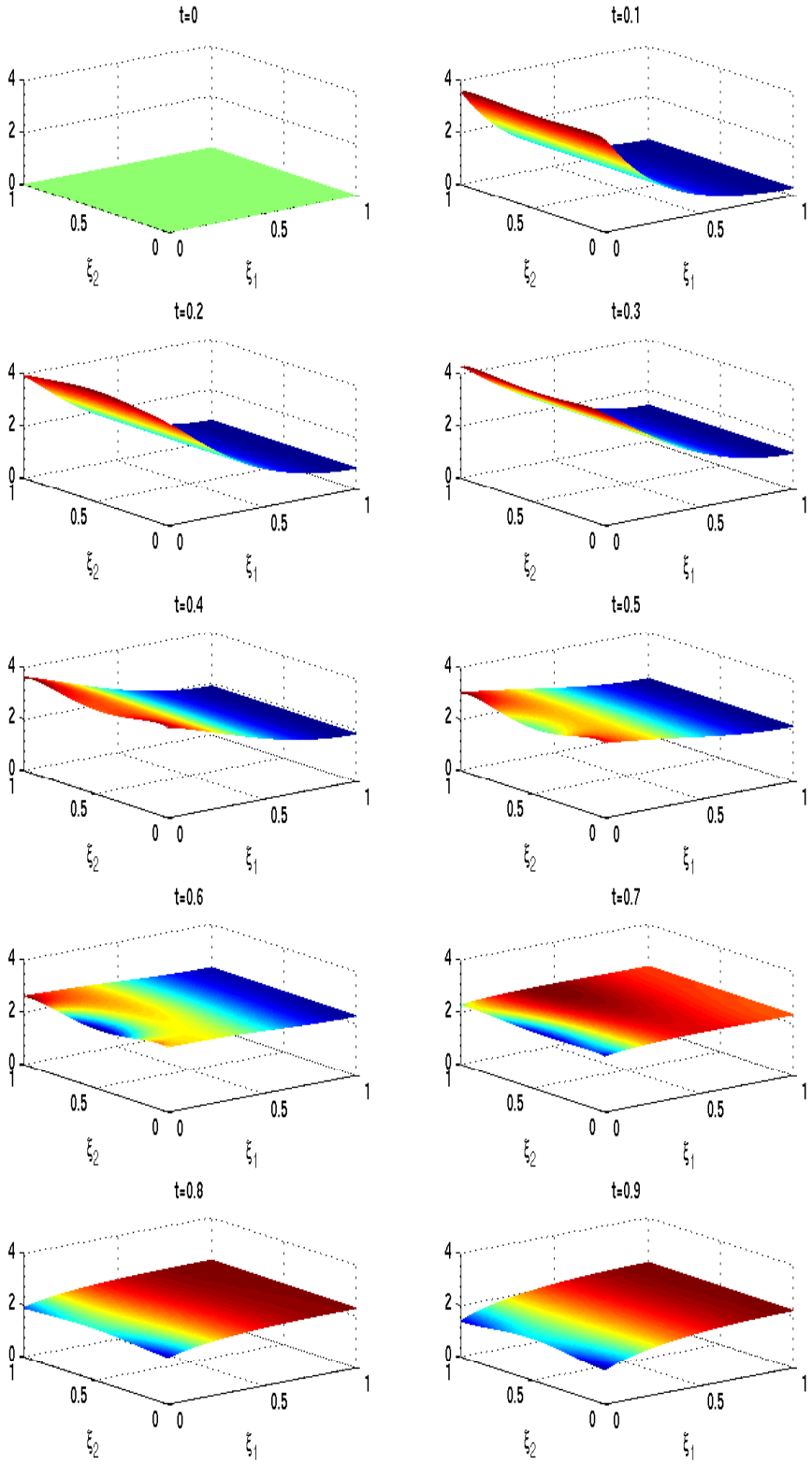


Figure 6.5: Snapshots of the optimal solution $\mathbf{x}(t, \xi)$.

Chapter 7

Conclusions and Future Work

In this work several iterative methods for the solution of large-scale matrix equations are proposed which are particularly adapted for the use in model order reduction as they compute the solutions in factorized format. The solvers exploit the structure of the underlying control problem by using $\mathcal{H}$-matrix approximations and are thus applicable to systems of large order when the coefficient matrices may be fully populated but allow for a data-sparse representation. The memory requirements of the $\mathcal{H}$-matrix based solvers are reduced from $\mathcal{O}(n^2)$ to linear-logarithmic size; the original cubic complexity of the iterative schemes is reduced to linear-polylogarithmic complexity. The performance (in terms of efficiency and accuracy) obtained by using the data-sparse solvers has been investigated in comparison with standard dense implementations of the iterative methods. It is observed by several numerical examples evolving from FEM/BEM discretizations of elliptic partial differential operators, that less memory is required, the computations are significantly faster and comparable accurate, when the developed methods are employed.

The solvers are used as basic building blocks for implementations of model reduction methods based on balanced truncation. Besides an approximate BT method, an implementation of SPA is described and a new method based on approximate low-rank factors of the cross-Gramian is proposed. We also derived an approach for model order reduction of unstable systems. All methods are of linear-polylogarithmic complexity and are thus suitable for reducing the dimensions of large-scale systems. The methods are successfully applied on systems of order $n = \mathcal{O}(10^5)$ coming from FEM and BEM discretizations of two- and three-dimensional parabolic PDEs also including varying diffusion coefficients or convective terms. Note that the application of balanced truncation model reduction to dense systems of this size becomes only possible by use of the data-sparse format and the corresponding formatted arithmetic. It is shown, both theoretically and in the numerical experiments, that the error between the original and the reduced-order system introduced by using the $\mathcal{H}$-matrix format and the approximate arithmetic is bounded. All methods significantly reduce the dimension of the systems, thereby still satisfying a very small error tolerance. Thus, reduced-order models of high quality are computed by computationally efficient implementations of BT and related methods. We demon-

strate how these reduced-order systems can be used in optimal control problems where inequality constraints for the control are given. Since the dimension of the resulting quadratic programming problem is significantly reduced, standard optimization software can be applied.

The application of the derived methods on further practically relevant problems should be investigated in the future. Moreover, further savings in computational time and memory requirements are expected when the following suggestions are considered.

1. All methods proposed in this work will benefit from further developments in the HLib [47].

   We expect a further decrease of the computational time and of the storage requirement in the iterative solvers if an alternative absolute criterion to the relative one (2.29) for the blockwise rank decision in the adaptive arithmetic would be provided, see Remark 4.1.2. This is announced for the next release of the HLib. Additional savings of memory and time are expected from an adaption of the blockwise accuracy ϵ during the iteration. For nearly converged A-iterates the formatted arithmetic can be performed with less accuracy resulting in smaller blockwise ranks.

   There are also several new data-sparse techniques available which are based on the hierarchical matrix format. Using a second hierarchy of nested bases besides the hierarchical block cluster tree (see Definition 2.3.4) defines the $\mathcal{H}^2$-matrices, see [42, 43, 48, 90] for details. This data-sparse format is suitable for discrete elliptic problems and for dense matrices arising from boundary element discretization of integral equations. Employing the technique to an $n \times n$ matrix will lead to memory requirements of $\mathcal{O}(nk)$ (instead of $\mathcal{O}(n \log_2(n)k)$ for $\mathcal{H}$-matrices) and to almost linear complexity in the corresponding arithmetic. Another efficient data-sparse construction for inverses of finite element stiffness matrices is proposed in [50]. These multilevel hierarchical matrices provide an efficient approximation of the inverses even for highly convection-dominated problems via an interpolation-based approach.

   Moreover, all iterative methods for the solution of matrix equations described in this work are composed of matrix computations which allow for parallel implementation. For parallel results of sign function and Smith iterations see, e.g., [31, 32, 35]. Once this feature is provided by the HLib, very large problems can be solved in significantly lower time on computers with distributed memory architecture. Approaches of improved efficiency based on special domain decomposition techniques are published in [80] and result in parallelizable $\mathcal{H}$-LU factorizations. Note that a parallel implementation of the hierarchical format and arithmetic is already available in the commercial release of the HLib, the $\mathcal{H}$-Lib$^{\text{pro}}$ [87]. This library is developed by the Max-Planck-Institute MIS in Leipzig and by Fraunhofer SCAI and is primarily designed for commercial and industrial applications.

2. The proposed methods for model order reduction are based on the hierarchical approximation of the system matrix A. Many practical applications lead to large-scale systems with this property and among them those problems would benefit most from the developed solvers where the resulting system is fully populated. Note that simulation and control of such systems is only possible in reasonable time if the dimension is significantly reduced by some order reduction approach. Till now, it was not possible to reduce the size of a large-scale, dense system by methods with system-theoretical background. In Section 5.1.3, the approximate BT method was successfully applied to a dense system of high order which stems from the BEM discretization of the Laplacian. The following (dense) problems could further be investigated in the future. Many physical laws, e.g. Maxwell's equation, can be described by LTI systems. For instance in chip design, a three-dimensional resistor-inductor-capacitor (RLC) interconnect is modeled by a spatial discretization of Maxwell's equation [124]. The resulting system is large-scale ($n \approx \mathcal{O}(10^6)$) and dense but allows for a data-sparse representation. Since the involved matrices are complex-valued which is not included in the HLib, an application of the proposed methods is a future task. Note that complex matrices are already incorporated in the library $\mathcal{H}$-Lib$^{\text{pro}}$. Other possible applications are boundary value problems discretized by wavelet techniques or combinations of finite element and boundary elements [148]. BEMs for parabolic problems are described in [56, 142].

3. The results in Section 6.2 indicate further possible application of the derived model order reduction methods in the area of nonlinear constrained optimal control. In a recent work [58], balanced truncation is used to reduce the dimension of the semi-discretized adjoint equation in optimal control problems governed by nonlinear PDEs. This provides an efficient evaluation of the adjoint equation in a descent method for the solution of the semi-discretized optimal control problem. Using sequential programming (SQP) methods for the solution of nonlinear constrained optimal control problems requires the solution of a linear quadratic programming problem in each iteration step. Applying model order reduction in the innermost loops would reduce memory requirement and computational time. The integration of the approximate BT method in optimal control problems offers many starting points for future research.

Bibliography

[1] R.W. Aldhaheri. Model order reduction via real Schur-form decomposition. *Internat. J. Control*, 53(3):709–716, 1991.

[2] A.C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM Publications, Philadelphia, PA, 2005.

[3] A.C. Antoulas, D.C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. *Contemp. Math.*, 280:193–219, 2001.

[4] A.C. Antoulas, D.C. Sorensen, and Y. Zhou. On the decay rate of Hankel singular values and related issues. *Sys. Control Lett.*, 46(5):323–342, 2002.

[5] R.M. Badía, P. Benner, R. Mayo, and E.S. Quintana-Ortí. Solving large sparse Lyapunov equations on parallel computers. In B. Monien and R. Feldmann, editors, *Euro-Par 2002 – Parallel Processing*, number 2400 in Lecture Notes in Computer Science, pages 687–690. Springer-Verlag, Berlin, Heidelberg, New York, 2002.

[6] R.M. Badía, P. Benner, R. Mayo, E.S. Quintana-Ortí, G. Quintana-Ortí, and A. Remón. Balanced truncation model reduction of large and sparse generalized linear systems. Technical Report Chemnitz Scientific Computing Preprints 06-04, Fakultät für Mathematik, TU Chemnitz, November 2006.

[7] Z. Bai. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Appl. Numer. Math*, 43(1–2):9–44, 2002.

[8] Z. Bai and J. Demmel. Design of a parallel nonsymmetric eigenroutine toolbox, Part I. In R.F. Sincovec et al., editor, *Proceedings of the Sixth SIAM Conference on Parallel Processing for Scientific Computing*, pages 391–398. SIAM, Philadelphia, PA, 1993.

[9] Z. Bai and J. Demmel. Design of a parallel nonsymmetric eigenroutine toolbox, Part II. Technical report, Computer Science Division, University of California, Berkeley, CA 94720, 1994.

[10] Z. Bai, P. Feldmann, and R.W. Freund. How to make theoretically passive reduced-order models passive in practice. In *Proceedings of the IEEE 1998 Custom Integrated Circuits Conference*, pages 207–210. IEEE, 1998.

[11] Z. Bai and R.W. Freund. A partial Padé-via-Lanczos method for reduced-order modeling. In *Proceedings of the Eighth Conference of the International Linear Algebra Society (Barcelona, 1999)*, volume 332/334, pages 139–164, 2001.

[12] Z. Bai, R.D. Slone, W.T. Smith, and Q. Ye. Error bound for reduced system model by Padé approximation via the Lanczos process. *IEEE Trans. CAD*, 18(2):133–141, 1999.

[13] A.Y. Barraud. A numerical algorithm to solve $A^T X A - X = Q$. *IEEE Trans. Automat. Control*, AC-22:883–885, 1977.

[14] R. H. Bartels and G. W. Stewart. Solution of the matrix equation $AX + XB = C$: Algorithm 432. *Comm. ACM*, 15:820–826, 1972.

[15] U. Baur. Low Rank Solution of Data-Sparse Sylvester Equations. Preprint #266, MATHEON, Berlin, FRG, `http://www.matheon.de`, October 2005. To appear in Numer. Lin. Alg. w. Appl.

[16] U. Baur and P. Benner. Factorized solution of the Lyapunov equation by using the hierarchical matrix arithmetic. *Proc. Appl. Math. Mech.*, 4(1):658–659, 2004.

[17] U. Baur and P. Benner. Factorized solution of Lyapunov equations based on hierarchical matrix arithmetic. *Computing*, 78(3):211–234, 2006.

[18] U. Baur and P. Benner. Gramian-based model reduction for data-sparse systems. Technical Report CSC/07-01, Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, FRG, 2007.

[19] A.N. Beavers and E. D. Denman. The matrix sign function and computations in systems. *Appl. Math. Comput.*, 2(1):63–94, 1976.

[20] M. Bebendorf. Efficient inversion of the Galerkin matrix of general second-order elliptic operators with nonsmooth coefficients. *Math. Comp.*, 74(251):1179–1199 (electronic), 2005.

[21] M. Bebendorf. Why approximate LU decompositions of finite element discretizations of elliptic operators can be computed with almost linear complexity. Preprint 8/2005, Max-Planck Institut für Mathematik in den Naturwissenschaften, Leipzig, Germany, 2005. To appear in SIAM J. Numer. Anal.

[22] M. Bebendorf and W. Hackbusch. Existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients. *Numer. Math.*, 95(1):1–28, 2003.

[23] P. Benner. Factorized solution of Sylvester equations with applications in control. In *Proc. Intl. Symp. Math. Theory Networks and Syst. MTNS 2004*, `http://www.mtns2004.be`, 2004.

[24] P. Benner. Computing Low-Rank Solutions of Algebraic Bernoulli Equations. Technical report, Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, FRG, November 2007.

[25] P. Benner, J.M. Claver, and E.S. Quintana-Ortí. Efficient solution of coupled Lyapunov equations via matrix sign function iteration. In A. Dourado et al., editor, *Proc.* 3rd *Portuguese Conf. on Automatic Control CONTROLO'98,* Coimbra, pages 205–210, 1998.

[26] P. Benner, J.M. Claver, and E.S. Quintana-Ortí. Parallel distributed solvers for large stable generalized Lyapunov equations. *Parallel Processing Letters*, 9(1):147–158, 1999.

[27] P. Benner, V. Mehrmann, V. Sima, S. Van Huffel, and A. Varga. SLICOT - a subroutine library in systems and control theory. In B.N. Datta, editor, *Applied and Computational Control, Signals, and Circuits*, volume 1, chapter 10, pages 499–539. Birkhäuser, Boston, MA, 1999.

[28] P. Benner, V. Mehrmann, and D. Sorensen, editors. *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin/Heidelberg, Germany, 2005.

[29] P. Benner and E.S. Quintana-Ortí. Model reduction based on spectral projection methods. Chapter 1 (pages 5–48) of [28].

[30] P. Benner and E.S. Quintana-Ortí. Solving stable generalized Lyapunov equations with the matrix sign function. *Numer. Algorithms*, 20(1):75–100, 1999.

[31] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. A portable subroutine library for solving linear control problems on distributed memory computers. In G. Cooperman, E. Jessen, and G.O. Michler, editors, *Workshop on Wide Area Networks and High Performance Computing, Essen (Germany), September 1998*, Lecture Notes in Control and Information, pages 61–88. Springer-Verlag, Berlin/Heidelberg, Germany, 1999.

[32] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Balanced truncation model reduction of large-scale dense systems on parallel computers. *Math. Comput. Model. Dyn. Syst.*, 6(4):383–405, 2000.

[33] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Singular perturbation approximation of large, dense linear systems. In *Proc. 2000 IEEE Intl. Symp. CACSD, Anchorage, Alaska, USA, September 25–27, 2000*, pages 255–260. IEEE Press, Piscataway, NJ, 2000.

[34] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Efficient numerical algorithms for balanced stochastic truncation. *Int. J. Appl. Math. Comp. Sci.*, 11(5):1123–1150, 2001.

[35] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Numerical solution of discrete stable linear matrix equations on multicomputers. *Parallel Algorithms and Appl.*, 17(1):127–146, 2002.

[36] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. State-space truncation methods for parallel model reduction of large-scale systems. *Parallel Comput.*, 29:1701–1722, 2003.

[37] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Parallel model reduction of large-scale linear descriptor systems via Balanced Truncation. In *High Performance Computing for Computational Science. Proc. 6th Intl. Meeting VECPAR'04, June 28–30, 2004, Valencia, Spain*, pages 65–78, 2004.

[38] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Solving stable Sylvester equations via rational iterative schemes. *J. Sci. Comp.*, 28(1):51–83, 2006.

[39] J.T. Betts and S.L. Campbell. Discretize then optimize. In *Mathematics for industry: challenges and frontiers*, pages 140–157. SIAM, Philadelphia, PA, 2005.

[40] C.H. Bischof and G. Quintana-Ortí. Algorithm 782: codes for rank-revealing QR factorizations of dense matrices. *ACM Trans. Math. Software*, 24(2):254–257, 1998.

[41] C.H. Bischof and G. Quintana-Ortí. Computing rank-revealing QR factorizations of dense matrices. *ACM Trans. Math. Software*, 24(2):226–253, 1998.

[42] S. Börm. $\mathcal{H}^2$-matrices—multilevel methods for the approximation of integral operators. *Comput. Vis. Sci.*, 7(3-4):173–181, 2004.

[43] S. Börm. $\mathcal{H}^2$-matrix arithmetics in linear complexity. *Computing*, 77(1):1–28, 2006.

[44] S. Börm. Approximation of solution operators of elliptic partial differential equations by $\mathcal{H}$- and $\mathcal{H}^2$-matrices. Preprint 85/2007, Max Planck Institute for Mathematics in the Sciences, 2007.

[45] S. Börm and L. Grasedyck. Hybrid cross approximation of integral operators. *Numer. Math.*, 101(2):221–249, 2005.

[46] S. Börm, L. Grasedyck, and W. Hackbusch. Introduction to hierarchical matrices with applications. *Engineering Analysis with Boundary Elements*, 27(5):405–422, May 2003.

[47] S. Börm, L. Grasedyck, and W. Hackbusch. HLib 1.3, 2004. Available from `http://www.hlib.org`.

[48] S. Börm and W. Hackbusch. Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices. *Computing*, 69(1):1–35, 2002.

[49] S. Le Borne. $\mathcal{H}$-matrices for convection-diffusion problems with constant convection. *Computing*, 70(3):261–274, 2003.

[50] S. Le Borne. Multilevel Hierarchical Matrices. *SIAM J. Matrix Anal. Appl.*, 28(3):871–889, 2006.

[51] S. Le Borne and L. Grasedyck. $\mathcal{H}$-matrix preconditioners in convection-dominated problems. *SIAM J. Matrix Anal. Appl.*, 27(4):1172–1183, 2006.

[52] R. Byers. Solving the algebraic Riccati equation with the matrix sign function. *Linear Algebra Appl.*, 85:267–279, 1987.

[53] R. Byers, C. He, and V. Mehrmann. The matrix sign function method and the computation of invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 18(3):615–632, 1997.

[54] T. Chan. Rank revealing QR factorizations. *Linear Algebra Appl.*, 88/89:67–82, 1987.

[55] S. Chandrasekaran and I. Ipsen. On rank-revealing factorisations. *SIAM J. Matrix Anal. Appl.*, 15(2):592–622, 1994.

[56] M. Costabel. Developments in boundary element methods for time-dependent problems. In L. Jentsch and F. Tröltzsch, editors, *Problems and methods in mathematical physics.*, pages 17–32. B. G. Teubner, 1994.

[57] L. Dai. *Singular Control Systems*. Number 118 in Lecture Notes in Control and Information Sciences. Springer-Verlag, Berlin, 1989.

[58] J.C. de los Reyes and T. Stykel. A balanced truncation based strategy for the optimal control of evolution problems. Manuscript, 2006.

[59] C. Durazzi. Numerical solution of discrete quadratic optimal control problems by Hestenes' method. In *Proceedings of the Workshop "Numerical Methods in Optimization" (Cortona, 1997)*, number 58, pages 133–154. Rend. Circ. Mat. Palermo (2) Suppl., 1999.

[60] D. Enns. Model reduction with balanced realization: an error bound and a frequency weighted generalization. In *Proc. 23rd IEEE Conf. Decis. Control*, pages 127–132, Las Vegas, NV, December 1984.

[61] W.H. Enright. Improving the efficiency of matrix operations in the numerical solution of stiff ordinary differential equations. *ACM Trans. Math. Softw.*, 4:127–136, 1978.

[62] P. Feldmann and R.W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. In *Proceedings of EURO-DAC '94 with EURO-VHDL '94, Grenoble, France*, pages 170–175. IEEE Computer Society Press, 1994.

[63] P. Feldmann and R.W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 14:639–649, 1995.

[64] K.V. Fernando and H. Nicholson. On the structure of balanced and other principal representations of SISO systems. *IEEE Trans. Automat. Control*, 28(2):228–231, 1983.

[65] K.V. Fernando and H. Nicholson. On a fundamental property of the cross-Gramian matrix. *IEEE Trans. Circuits Syst.*, CAS-31(5):504–505, 1984.

[66] R.W. Freund. Model reduction methods based on Krylov subspaces. *Acta Numerica*, 12:267–319, 2003.

[67] K. Gallivan, E. Grimme, and P. Van Dooren. Asymptotic waveform evaluation via a Lanczos method. *Appl. Math. Lett.*, 7(5):75–80, 1994.

[68] K. Gallivan, A. Vandendorpe, and P. Van Dooren. Sylvester equations and projection-based model reduction. In *Proceedings of the International Conference on Linear Algebra and Arithmetic (Rabat, 2001)*, volume 162, pages 213–229, 2004.

[69] J.D. Gardiner and A.J. Laub. A generalization of the matrix-sign-function solution for algebraic Riccati equations. *Internat. J. Control*, 44:823–832, 1986.

[70] J.D. Gardiner, A.J. Laub, J.J. Amato, and C.B. Moler. Solution of the Sylvester matrix equation $AXB + CXD = E$. *ACM Trans. Math. Software*, 18:223–231, 1992.

[71] W. Gawronski and J.-N. Juang. Model reduction in limited time and frequency intervals. *Internat. J. Systems Sci.*, 21(2):349–376, 1990.

[72] K. Glover. All optimal Hankel-norm approximations of linear multivariable systems and their L^∞ norms. *Internat. J. Control*, 39:1115–1193, 1984.

[73] G.H. Golub, S. Nash, and C.F. Van Loan. A Hessenberg–Schur method for the problem $AX + XB = C$. *IEEE Trans. Automat. Control*, AC-24:909–913, 1979.

[74] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996.

[75] L. Grasedyck. *Theorie und Anwendungen Hierarchischer Matrizen*. Dissertation, University of Kiel, Kiel, Germany, 2001. In German, available at `http://e-diss.uni-kiel.de/diss_454`.

[76] L. Grasedyck. Existence of a low rank or H-matrix approximant to the solution of a Sylvester equation. *Numer. Lin. Alg. Appl.*, 11:371–389, 2004.

[77] L. Grasedyck and W. Hackbusch. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing*, 70(4):295–334, 2003.

[78] L. Grasedyck and W. Hackbusch. A multigrid method to solve large scale Sylvester equations. *SIAM J. Matrix Anal. Appl.*, 29(3):870–894, 2007.

[79] L. Grasedyck, W. Hackbusch, and B. N. Khoromskij. Solution of large scale algebraic matrix Riccati equations by use of hierarchical matrices. *Computing*, 70:121–165, 2003.

[80] L. Grasedyck, R. Kriemann, and S. Le Borne. Parallel black box domain decomposition based $\mathcal{H}$-LU preconditioning. Preprint 115/2005, Max-Planck Institut für Mathematik in den Naturwissenschaften, Leipzig, Germany, 2005.

[81] E.J. Grimme. *Krylov projection methods for model reduction*. PhD thesis, Univ. Illinois, Urbana-Champaign, 1997.

[82] E.J. Grimme, D.C. Sorensen, and P. Van Dooren. Model reduction of state space systems via an implicitly restarted Lanczos method. *Numer. Algorithms*, 12:1–31, 1996.

[83] S. Gugercin and A.C. Antoulas. A survey of model reduction by balanced truncation and some new results. *Internat. J. Control*, 77(8):748–766, 2004.

[84] S. Gugercin, A.C. Antoulas, and C. Beattie. $\mathcal{H}_2$ Model Reduction for large-scale dynamical systems. *SIAM J. Matrix Anal. Appl.*, to appear.

[85] S. Gugercin and J.-R. Li. Smith-type methods for balanced truncation of large systems. Chapter 2 (pages 49–82) of [28].

[86] S. Gugercin, D.C. Sorensen, and A.C. Antoulas. A modified low-rank Smith method for large-scale Lyapunov equations. *Numer. Algorithms*, 32(1):27–55, 2003.

[87] H-Lib$^{\text{pro}}$, `http://www.scai.fraunhofer.de/hlibpro.html`.

[88] W. Hackbusch. *Integral equations. Theory and Numerical Treatment*, volume 120 of *International Series of Numerical Mathematics*. Birkhäuser Verlag, Basel, 1995.

[89] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

[90] W. Hackbusch, B. Khoromskij, and S. A. Sauter. On $\mathcal{H}^2$-matrices. In *Lectures on applied mathematics (Munich, 1999)*, pages 9–29. Springer, Berlin, 2000.

[91] W. Hackbusch and B. N. Khoromskij. A sparse $\mathcal{H}$-matrix arithmetic. II. Application to multi-dimensional problems. *Computing*, 64(1):21–47, 2000.

[92] S.J. Hammarling. Numerical solution of the stable, non-negative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2:303–323, 1982.

[93] S.J. Hammarling. Numerical solution of the discrete-time, convergent, non-negative definite Lyapunov equation. *Sys. Control Lett.*, 17:137–139, 1991.

[94] HAPACK, `http://www.tu-chemnitz.de/mathematik/hapack/`.

[95] G.A. Hewer and C. Kenney. The sensitivity of the stable Lyapunov equation. *SIAM J. Cont. Optim.*, 26:321–344, 1988.

[96] N.J. Higham. Computing the polar decomposition—with applications. *SIAM J. Sci. Statist. Comput.*, 7:1160–1174, 1986.

[97] N.J. Higham. *Accuracy and stability of numerical algorithms*. SIAM Publications, Philadelphia, PA, second edition, 2002.

[98] D. Hinrichsen and A.J. Pritchard. *Mathematical systems theory. I – Modelling, state space analysis, stability and robustness*, volume 48 of *Texts in Applied Mathematics*. Springer-Verlag, Berlin, 2005.

[99] M. Hochbruck and G. Starke. Preconditioned Krylov subspace methods for Lyapunov matrix equations. *SIAM J. Matrix Anal. Appl.*, 16(1):156–171, 1995.

[100] A.S. Hodel, B. Tenison, and K.R. Poolla. Numerical solution of the Lyapunov equation by approximate power iteration. *Linear Algebra Appl.*, 236:205–230, 1996.

[101] D.Y. Hu and L. Reichel. Krylov-subspace methods for the Sylvester equation. *Linear Algebra Appl.*, 172:283–313, 1992.

[102] I.M. Jaimoukha and E.M. Kasenally. Krylov subspace methods for solving large Lyapunov equations. *SIAM J. Numer. Anal.*, 31:227–251, 1994.

[103] I.M. Jaimoukha and E.M. Kasenally. Implicitly restarted Krylov subspace methods for stable partial realizations. *SIAM J. Matrix Anal. Appl.*, 18(3):633–652, 1997.

[104] K. Jbilou and A.J. Riquet. Projection methods for large Lyapunov matrix equations. *Linear Algebra Appl.*, 415(2–3):344–358, 2006.

[105] I. Jonsson and B. Kågström. Recursive blocked algorithms for solving triangular systems. I: One-sided and coupled Sylvester-type matrix equations. *ACM Trans. Math. Software*, 28(4):392–415, 2002.

[106] C. Kenney and A.J. Laub. On scaling Newton's method for polar decomposition and the matrix sign function. *SIAM J. Matrix Anal. Appl.*, 13:688–706, 1992.

[107] C. Kenney and A.J. Laub. The matrix sign function. *IEEE Trans. Automat. Control*, 40(8):1330–1348, 1995.

[108] D. Kressner. *Numerical methods for general and structured eigenvalue problems*, volume 46 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2005.

[109] D. Kressner. Block variants of Hammarling's method for solving Lyapunov equations. Technical report, Department of Computing Science, Umea University, Sweden, July 2006. Revised January 2007. Available from `http://www.cs.umu.se/~kressner/pubs.php`. To appear in ACM Trans. Math. Software.

[110] P. Kunkel and V. Mehrmann. *Differential-algebraic equations – Analysis and numerical solution*. EMS Textbooks in Mathematics. European Mathematical Society (EMS), Zürich, 2006.

[111] P. Lancaster. Explicit solutions of linear matrix equations. *SIAM Rev.*, 12:544–566, 1970.

[112] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford University Press, Oxford, 1995.

[113] P. Lancaster and M. Tismenetsky. *The Theory of Matrices*. Academic Press, Orlando, 2nd edition, 1985.

[114] V.B. Larin and F.A. Aliev. Construction of square root factor for solution of the Lyapunov matrix equation. *Sys. Control Lett.*, 20:109–112, 1993.

[115] A.J. Laub, M.T. Heath, C.C. Paige, and R.C. Ward. Computation of system balancing transformations and other application of simultaneous diagonalization algorithms. *IEEE Trans. Automat. Control*, 34:115–122, 1987.

[116] A.J. Laub, L.M. Silverman, and M. Verma. A note on cross-Grammians for symmetric realizations. *Proceedings of the IEEE Trans. Circuits Syst.*, 71(7):904–905, 1983.

[117] J.-R. Li, F. Wang, and J. White. An efficient Lyapunov equation-based approach for generating reduced-order models of interconnect. *In Proc. Design Automation Conference*, pages 1–6, 1999.

[118] J.-R. Li and J. White. Reduction of large circuit models via low rank approximate gramians. *Int. J. Appl. Math. Comp. Sci.*, 11(5):1151–1171, 2001.

[119] J.-R. Li and J. White. Low rank solution of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 24(1):260–280, 2002.

[120] M. Lintner. The eigenvalue problem for the 2D Laplacian in $\mathcal{H}$-matrix arithmetic and application to the heat and wave equation. *Computing*, 72(3-4):293–323, 2004.

[121] W.Q. Liu and V. Sreeram. Model reduction of singular systems. In *Proc.39th IEEE Conf. Dec. Control 2000*, pages 2373–2378, 2000.

[122] Y. Liu and B.D.O. Anderson. Controller reduction via stable factorization and balancing. *Internat. J. Control*, 44:507–531, 1986.

[123] A. Lu and E.L. Wachspress. Solution of Lyapunov equations by alternating direction implicit iteration. *J. Comput. Appl. Math.*, 21:43–58, 1991.

[124] N.A. Marques, M. Kamon, L.M. Silveira, and J.K. White. Generating compact, guaranteed passive reduced-order models of 3-d RLC interconnects. *IEEE Trans. Advanced Packaging*, 27(4):569– 580, 2004.

[125] The MathWorks, Inc., `http://www.matlab.com`.

[126] V. Mehrmann and T. Stykel. Balanced truncation model reduction for large-scale systems in descriptor form. Chapter 3 (pages 83–115) of [28].

[127] B.C. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Trans. Automat. Control*, AC-26:17–32, 1981.

[128] The MOSEK Optimization Software, `http://www.mosek.com/`.

[129] C. Mullis and R.A. Roberts. Synthesis of minimum roundoff noise fixed point digital filters. *IEEE Trans. Circuits and Systems*, CAS-23(9):551–562, 1976.

[130] A.G.O. Mutambara. *Design and Analysis of Control Systems*. CRC Press, Boca Raton, FL, 1999.

[131] G. Obinata and B.D.O. Anderson. *Model Reduction for Control System Design*. Communications and Control Engineering Series. Springer-Verlag, London, UK, 2001.

[132] T. Penzl. A multi-grid method for generalized Lyapunov equations. Technical Report SFB393/97-24, Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, FRG, 1997. Available from `http://www.tu-chemnitz.de/sfb393/sfb97pr.html`.

[133] T. Penzl. Numerical solution of generalized Lyapunov equations. *Adv. Comp. Math.*, 8:33–48, 1997.

[134] T. Penzl. *Numerische Lösung großer Lyapunov-Gleichungen*. Logos–Verlag, Berlin, Germany, 1998. Dissertation, Fakultät für Mathematik, TU Chemnitz, 1998.

[135] T. Penzl. A cyclic low rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418, 2000.

[136] T. Penzl. Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. *Sys. Control Lett.*, 40:139–144, 2000.

[137] T. Penzl. LYAPACK Users Guide. Technical Report SFB393/00-33, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, FRG, 2000. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`.

[138] K. Perev and B. Shafai. Balanced realization and model reduction of singular systems. *Internat. J. Systems Sci.*, 25(6):1039–1052, 1994.

[139] L. Pernebo and L.M. Silverman. Model reduction via balanced state space representations. *IEEE Trans. Automat. Control*, 27(2):382–387, 1982.

[140] L.T. Pillage and R.A. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Trans. CAD*, 9:325–366, 1990.

[141] R. Plato. *Numerische Mathematik kompakt*. Vieweg, Wiesbaden, 2 edition, 2004.

[142] C. Pozrikidis. *A practical guide to boundary element methods with the software library BEMLIB*. Chapman & Hall/CRC, Boca Raton, FL, 2002.

[143] P. Rabiei and M. Pedram. Model order reduction of large circuits using balanced truncation. In *Proc. of Asia and South Pacific Design Automation Conf.*, pages 237–240, February 1999.

[144] J.D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32:677–687, 1980. (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971).

[145] I.G. Rosen and C. Wang. A multi–level technique for the approximate solution of operator Lyapunov and algebraic Riccati equations. *SIAM J. Numer. Anal.*, 32(2):514–541, 1995.

[146] Y. Saad. Numerical solution of large Lyapunov equations. In M. A. Kaashoek, J. H. van Schuppen, and A. C. M. Ran, editors, *Signal Processing, Scattering, Operator Theory and Numerical Methods*, pages 503–511. Birkhäuser, 1990.

[147] S.A. Sauter and C. Schwab. *Randelementmethoden*. B. G. Teubner, Stuttgart, Leipzig, Wiesbaden, 2004.

[148] R. Schneider. *Multiskalen- und Wavelet-Matrixkompression: Analysisbasierte Methoden zur effizienten Lösung großer vollbesetzter Gleichungssysteme*. B. G. Teubner, Stuttgart, Leipzig, Wiesbaden, 1998.

[149] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics*. Marcel Dekker, Inc., New York, NY, 1996.

[150] V. Simoncini. A new iterative method for solving large-scale Lyapunov matrix equations. *SIAM J. Sci. Comput.*, 29(3):1268–1288, 2007.

[151] SLICOT, `http://www.slicot.org`.

[152] R.A. Smith. Matrix equation $XA + BX = C$. *SIAM J. Appl. Math.*, 16(1):198–201, 1968.

[153] V. Sokolov. *Contributions to the Minimal Realization Problem for Descriptor Systems*. Dissertation, Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz (Germany), January 2006.

[154] E.D. Sontag. *Mathematical Control Theory*. Springer-Verlag, New York, NY, 2nd edition, 1998.

[155] D.C. Sorensen and A.C. Antoulas. Dimension reduction of large-scale systems. Chapter 4 (pages 117–130) of [28].

[156] D.C. Sorensen and A.C. Antoulas. The Sylvester equation and approximate balanced reduction. *Linear Algebra Appl.*, 351/352:671–700, 2002.

[157] D.C. Sorensen and Y. Zhou. Bounds on eigenvalue decay rates and sensitivity of solutions to Lyapunov equations. Technical Report TR02-07, Dept. of Comp. Appl. Math., Rice University, Houston, TX, June 2002. Available online from `http://www.caam.rice.edu/caam/trs/tr02.html#TR02-07`.

[158] T. Stykel. *Analysis and Numerical Solution of Generalized Lyapunov Equations*. Dissertation, TU Berlin, 2002.

[159] T. Stykel. Gramian-based model reduction for descriptor systems. *Math. Control Signals Systems*, 16(4):297–319, 2004.

[160] T. Stykel. Balanced truncation model reduction for semidiscretized Stokes equation. *Linear Algebra Appl.*, 415(2–3):262–289, 2006.

[161] M.S. Tombs and I. Postlethwaite. Truncated balanced realization of a stable non-minimal state-space system. *Internat. J. Control*, 46(4):1319–1330, 1987.

[162] F. Tröltzsch. *Optimal control of partial differential equations. Theory, procedures, and applications. (Optimale Steuerung partieller Differentialgleichungen. Theorie, Verfahren und Anwendungen.)*. Wiesbaden: Vieweg. x, 297 p., 2005.

[163] P. Van Dooren. Gramian based model reduction of large-scale dynamical systems. In D.F. Griffiths and G.A. Watson, editors, *Numerical Analysis 1999. Proc. 18th Dundee Biennial Conference on Numerical Analysis*, pages 231–247, London, UK, 2000. Chapman & Hall/CRC.

[164] A. Varga. A note on Hammarling's algorithm for the discrete Lyapunov equation. *Sys. Control Lett.*, 15(3):273–275, 1990.

[165] A. Varga. Efficient minimal realization procedure based on balancing. In *Prepr. of the IMACS Symp. on Modelling and Control of Technological Systems*, volume 2, pages 42–47, 1991.

[166] E.L. Wachspress. Iterative solution of the Lyapunov matrix equation. *Appl. Math. Letters*, 107:87–90, 1988.

[167] K. Zhou, G. Salomon, and E. Wu. Balanced realization and model reduction for unstable systems. *Int. J. Robust Nonlinear Control*, 9(3):183–198, 1999.