Informationsveranstaltung RW/CSE

Persönliche Erfahrungen aus dem Studium und Präsentation der Masterarbeit

Giuseppe Accaputo, 18.05.2017

Über mich

- 4. Semester RW/CSE Master
  - Vertiefungsgebiete: Physik und Chemie
- Lehre als Informatiker mit Berufsmatura
- Fachhochschulabschluss (Bachelor) in Informatik
- 2 Jahre Berufserfahrung

Warum RW/CSE?

- Wollte zuerst Mathematik studieren
- Wollte jedoch auch praxisnah bleiben
- Studienberatung an der ETH empfahl mir RW/CSE

Der RW/CSE Studiengang ist vielseitig

- Sehr interdisziplinär
  - Mix aus Mathematik, Informatik und Naturwissenschaften allgemein
- Bereits im Bachelor-Studium hat man Vorlesungen mit Bachelor und Master-Studenten aus anderen Fachrichtungen
- Viele Vertiefungsmöglichkeiten, wie z.B. Physik, Chemie, Biologie, Robotik, Finanzmathematik, etc.

Der RW/CSE Studiengang ist spannend

- Grosse Auswahl an interessanten Vorlesungen
  - Von *How To Write Fast Numerical Code* bis zu *Advanced Quantum Chemistry*
- Zugriff auf Supercomputer der ETH und des CSCS (Centro Svizzero di Calcolo Scientifico)
- High-Performance Computing (HPC) ist ein wichtiges Hilfsmittel
- Forschung sowie auch Wirtschaft setzen immer mehr auf Simulationen

Der RW/CSE Studiengang ist persönlich

- Kleiner Studiengang (ca. 20-30 Studenten pro Jahrgang)
- Direkter Kontakt zu Studienvorsitzende
- Eure Meinung zählt
  - Anpassungen dank Studenten-Feedback
- Ganz viele Apéros

Solving Large Scale Eigenvalue Problems in Amorphous Materials

Präsentation der Masterarbeit

Einführung: the Boson Peak

- Metallisches Glas ist amorphes Material; Atome verfügen über keine geordneten Strukturen
- Der Boson Peak: experimentelle Signatur von amorphen Materialien
  - Ziel: Herkunft des Boson Peaks analysieren
- Molekulardynamik-Simulationen (*Molecular Dynamics Simulations*) können metallische Glase simulieren / generieren

Einführung: the Boson Peak

- Genauere Analyse des Boson Peaks durch Diagonalisierung der Hesse-Matrix $\boldsymbol{H}$ (wird durch die MD-Simulation generiert)
- Die Hesse-Matrix $\boldsymbol{H} \in \mathbb{R}^{n \times n}$ (reell-symmetrisch) sollte jedoch nicht komplett diagonalisiert werden, da uns nur $q \ll n$ Eigenwerte und Eigenvektoren in der nähe des Boson Peaks interessieren
  - $n \approx 1.5 \cdot 10^9$
  - $q \approx 100$ bis zu 1000

Polynomialer Filter für Eigenwertprobleme [1]

Li, Ruipeng, et al. "A Thick-Restart Lanczos algorithm with polynomial filtering for Hermitian eigenvalue problems." *SIAM Journal on Scientific Computing* 38.4 (2016): A2512-A2534.

1. *Spectrum Slicing*
2. Polynomialer Filter
3. Berechnung der Eigenwerte und Eigenvektoren

Spectrum Slicing

- Sei I_σ das Intervall, welches das komplette Spektrum $\sigma(\boldsymbol{H})$ enthält:

$$I_\sigma = [\lambda_{min}, \lambda_{max}] \tag{1}$$

- Partitioniere I_σ in r Teilintervalle $s_1, s_2, ..., s_r$ so, dass

$$I_\sigma = s_1 \cup s_2 \cup \ldots \cup s_r \qquad s_i = [\xi_i, \eta_i] \tag{2}$$

- Verteile die Teilintervalle auf verschiedene Prozessoren / Nodes, sodass auf den Teilintervalle gleichzeitig Berechnungen durchgeführt werden können
  - *Load Balancing*: Spektrale Dichte kann approximiert werden [2]; wähle die Teilintervalle so, dass alle s_i in etwa gleich viele Eigenwerte enthalten

Polynomialer Filter

- *Least-Squares* Polynomialer Filter [1]: Methode um Eigenwerte in einem gegebenen Intervall zu filtern
- Wir möchten in einem Teilintervall nur die Eigenwerte berechnen, welche sich im Intervall befinden; Eigenwerte ausserhalb des Intervalls möchten wir ignorieren
- Definiere eine Filter-Funktion $\hat{\rho}_k(\lambda)$ und eine Threshold τ so, dass für Eigenwerte im Teilintervall $\hat{\rho}_k(\lambda) \geq \tau$ gilt und für Eigenwerte ausserhalb des Intervalls $\hat{\rho}_k(\lambda) < \tau$ gilt

Polynomialer Filter

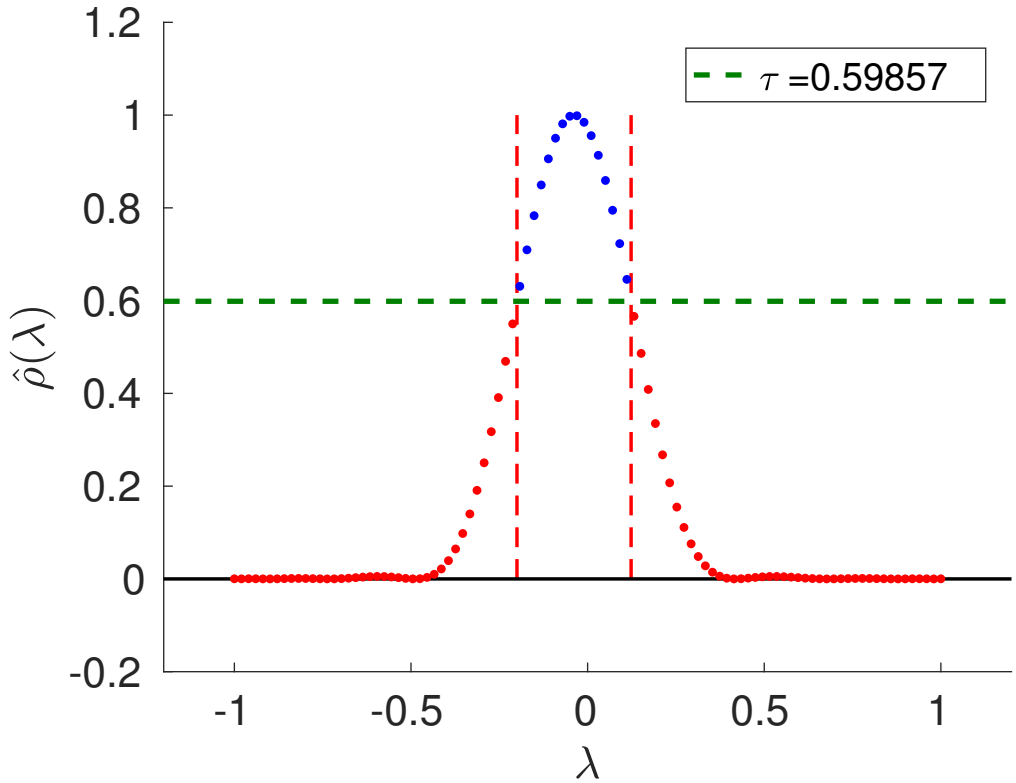

Figure 1: Filter $\hat{\rho}_k(\lambda)$ für das Intervall $[-0.2, 0.12312]$.
Rote Punkte: $\hat{\rho}(t) < \tau$; Blaue Punkte: $\hat{\rho}(t) \geq \tau$.

Polynomialer Filter

- Filter ist eine Approximation der Dirac-Delta Funktion und wird aus Chebyshev Polynomen konstruiert
- Optimaler Grad des Filters wird für jedes Intervall einzeln bestimmt / berechnet
- Entferne Oszillationen mittels Smoothers; Filter wird *geglättet*
- Filter muss nachträglich evtl. balanciert werden

Polynomialer Filter: Smoothing (*glätten*)

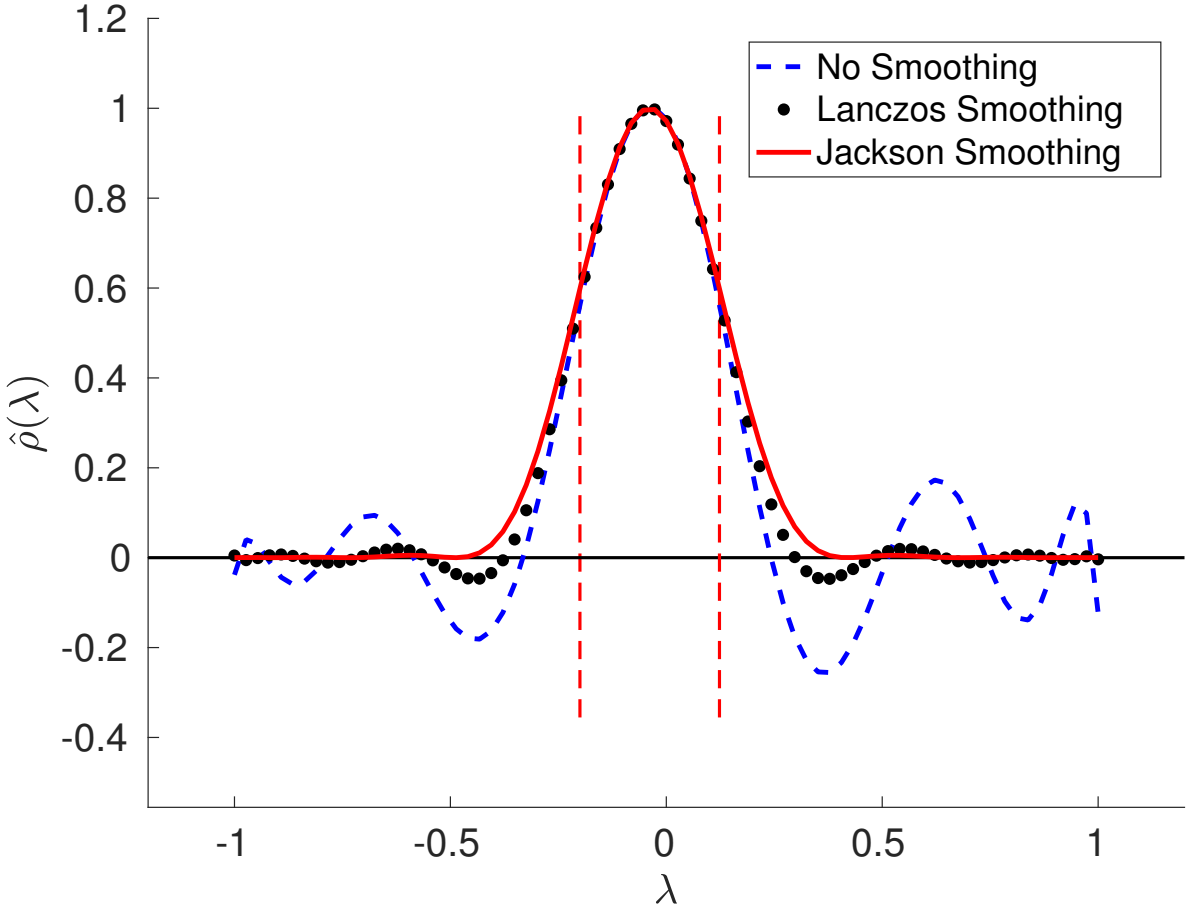

Figure 2: Filter $\hat{\rho}_k(\lambda)$ mit verschiedenen Smoothers.

Polynomialer Filter: Balancieren

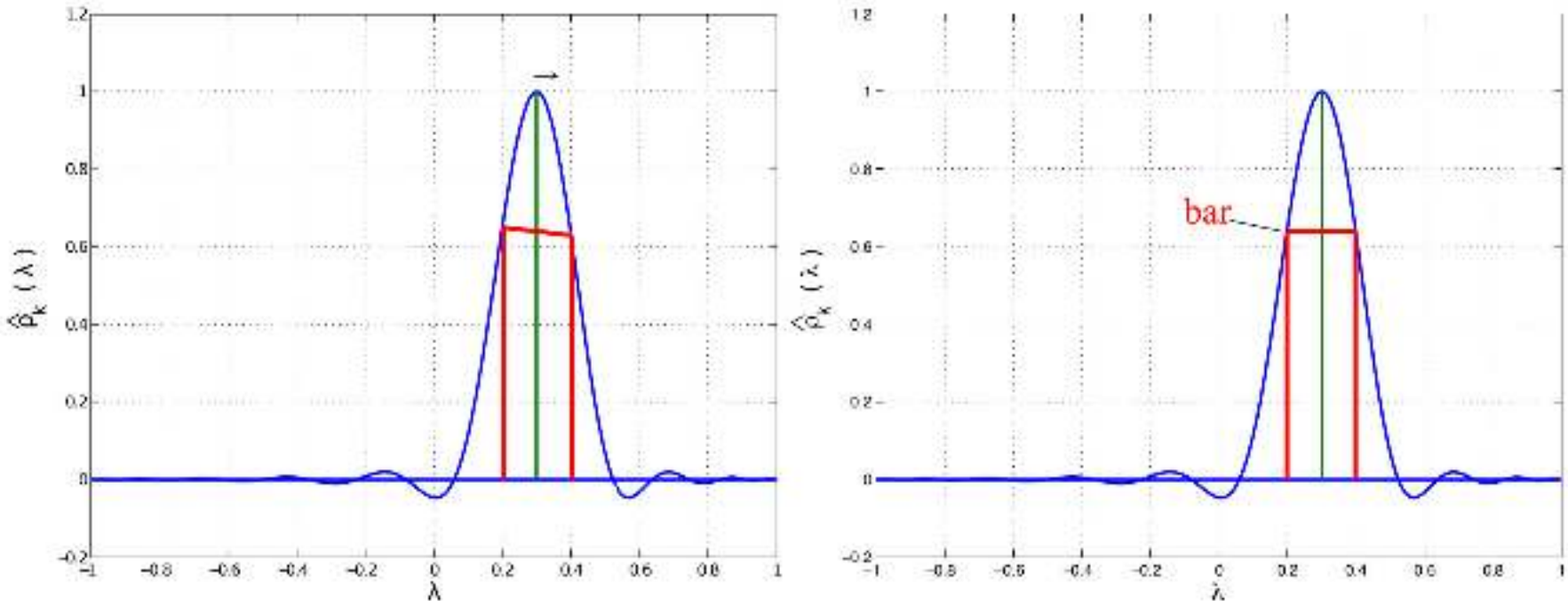

Figure 3: Balanciere Filter so, dass $\hat{\rho}_k(\xi_i) = \hat{\rho}_k(\eta_i) = \tau$ (bar)

Berechnung der Eigenwerte

- Eigenwerte werden mittels *thick-restarted Lanczos algorithm with deflation* berechnet
  - *Thick-restart*: Lanczos wird mit k Vektoren neu-gestartet; dabei sind diese Vektoren gute Approximationen (noch nicht konvergiert)
  - *Deflation*: Wenn ein Eigenpaar konvergiert ist, wird es aus dem Suchraum für die weiteren Iterationen entfernt

Polynomialer Filter: Visuelles Beispiel

- $\lambda_{\min} = 0, \lambda_{\max} = 8$
- $\sigma = \{\lambda_{\min},\ 2,\ 2.3,\ 2.6,\ 3,\ 3.4,\ 6.1,\ 6.4,\ 7.1,\ 7.5,\ 7.7,\ \lambda_{\max}\}$
  - $I_\sigma = [\lambda_{\min}, \lambda_{\max}]$
- Jackson smoothing
- Anzahl Teilintervalle = 4 (3 Eigenwerte per Teilintervall)
  - $s_1 = [\lambda_{\min}, 2.4]$
  - $s_2 = [2.5, 3.5]$
  - $s_3 = [6, 7.2]$
  - $s_4 = [7.4, \lambda_{\max}]$

Polynomialer Filter: Visuelles Beispiel

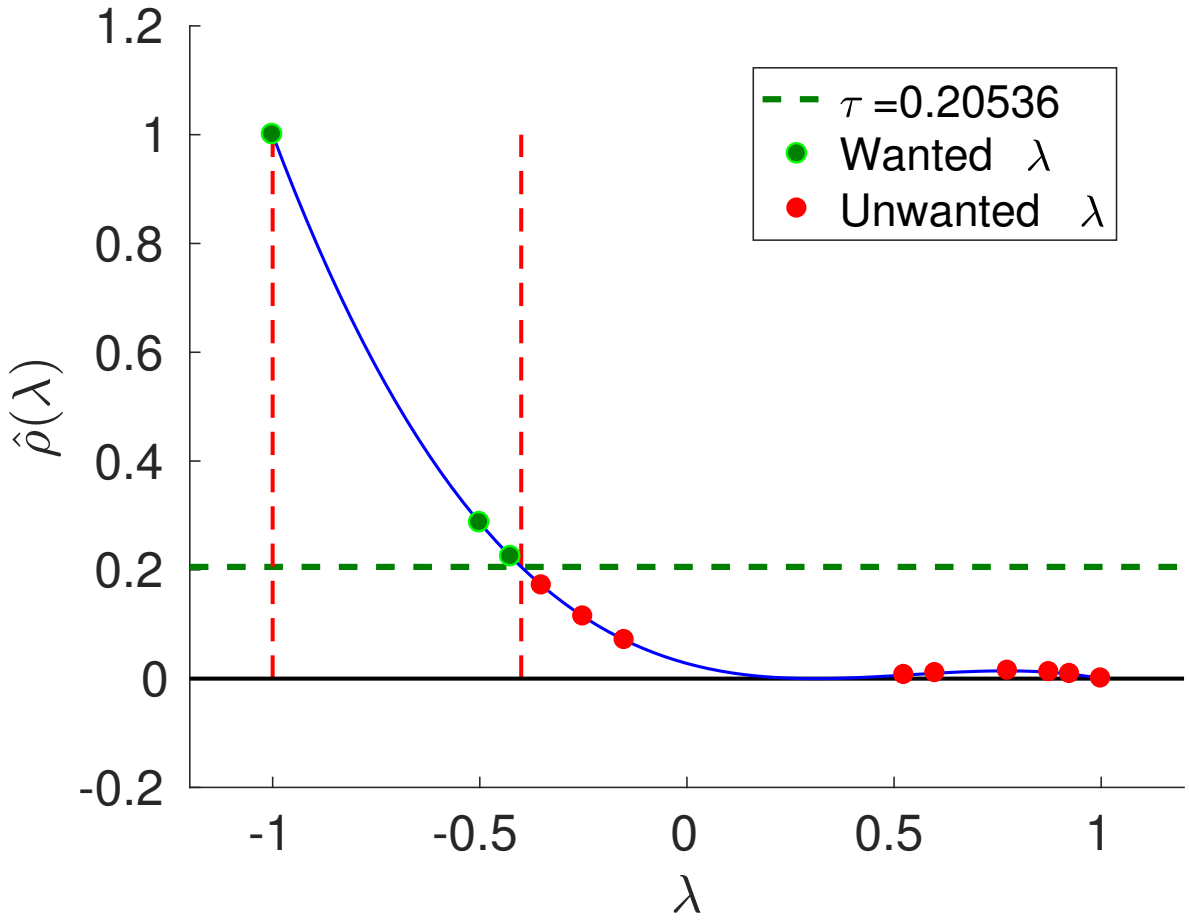

Figure 4: Filter $\hat{\rho}_k(\lambda)$ (Grad = 3) für das Intervall $s_1 = [\lambda_{\min}, 2.4]$.

Polynomialer Filter: Visuelles Beispiel

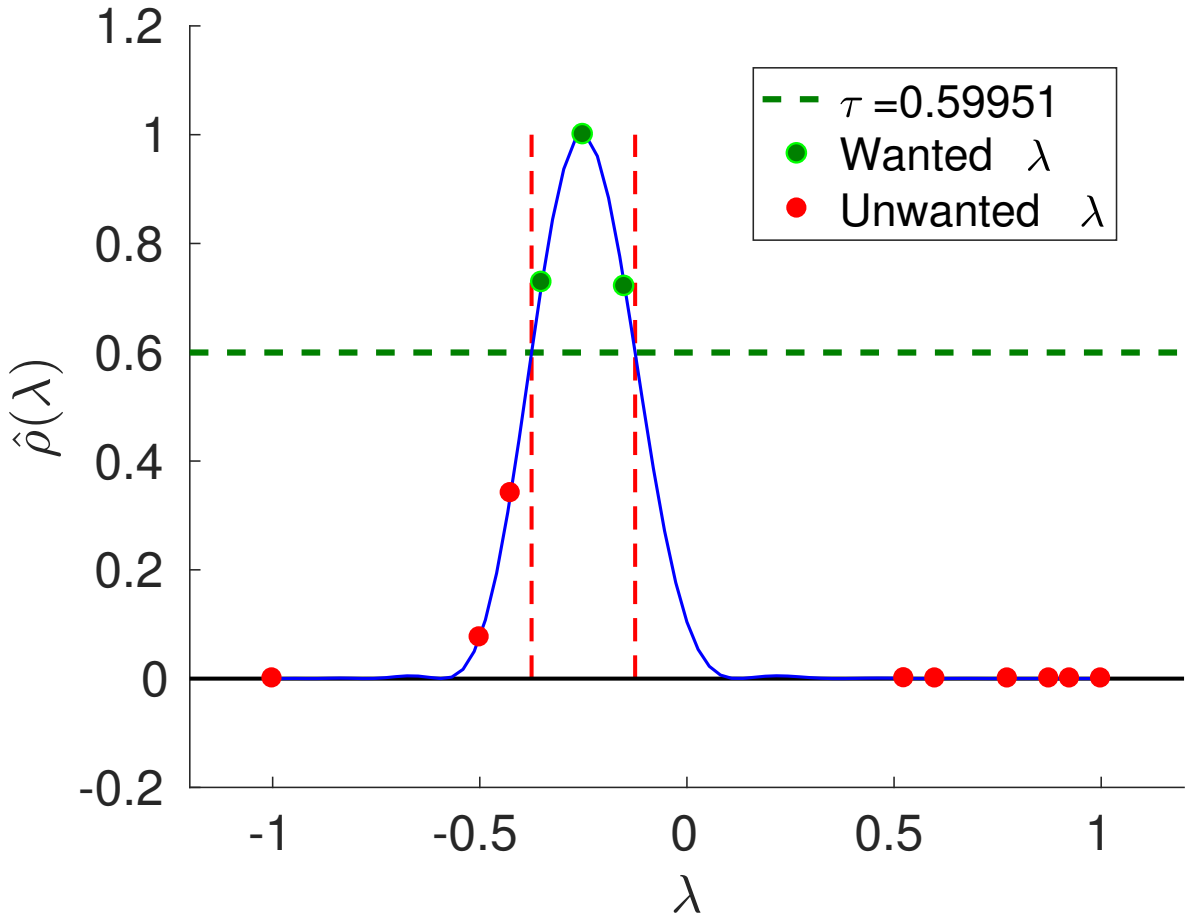

Figure 5: Filter $\hat{\rho}_k(\lambda)$ (Grad = 23) für das Intervall $s_2 = [2.5, 3.5]$

Polynomialer Filter: Visuelles Beispiel

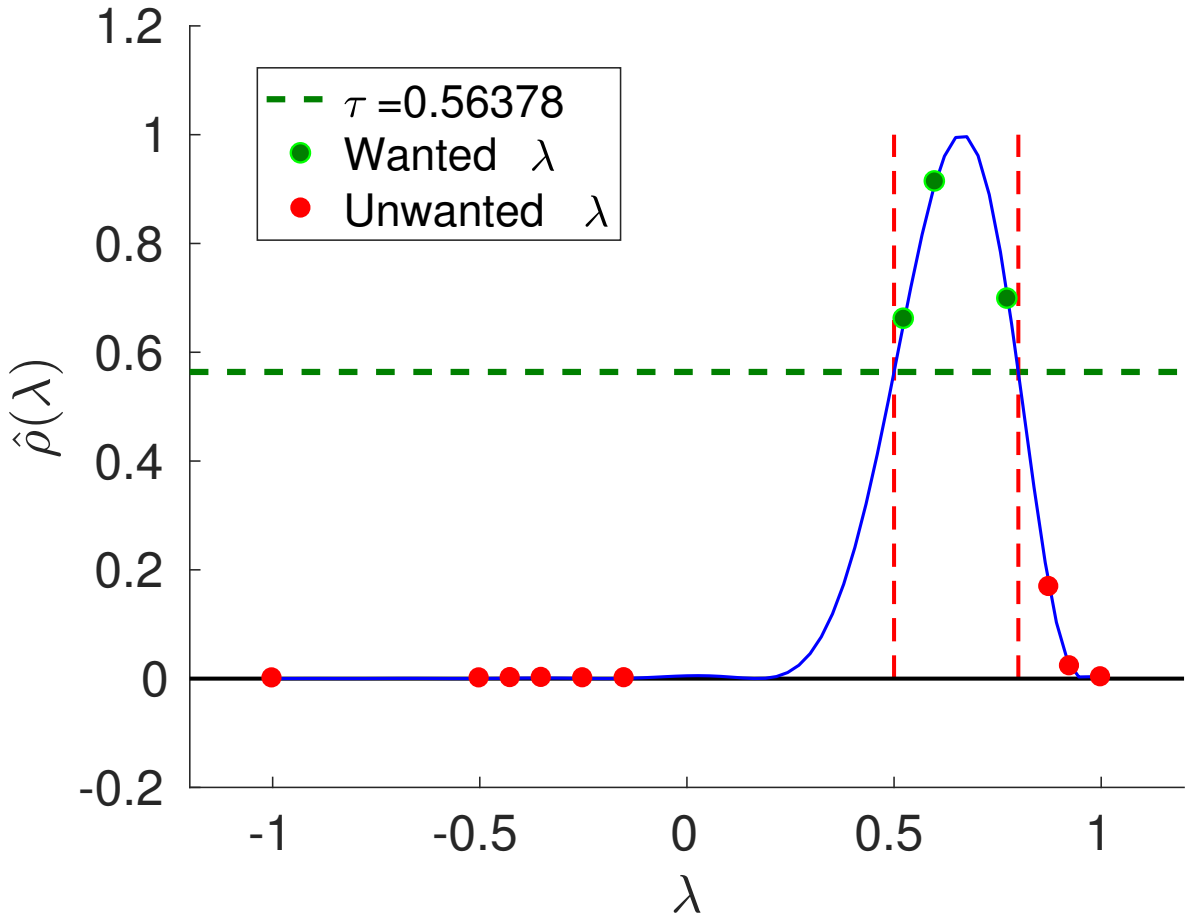

Figure 6: Filter $\hat{\rho}_k(\lambda)$ (Grad = 15) für das Intervall $s_3 = [6, 7.2]$

Polynomialer Filter: Visuelles Beispiel

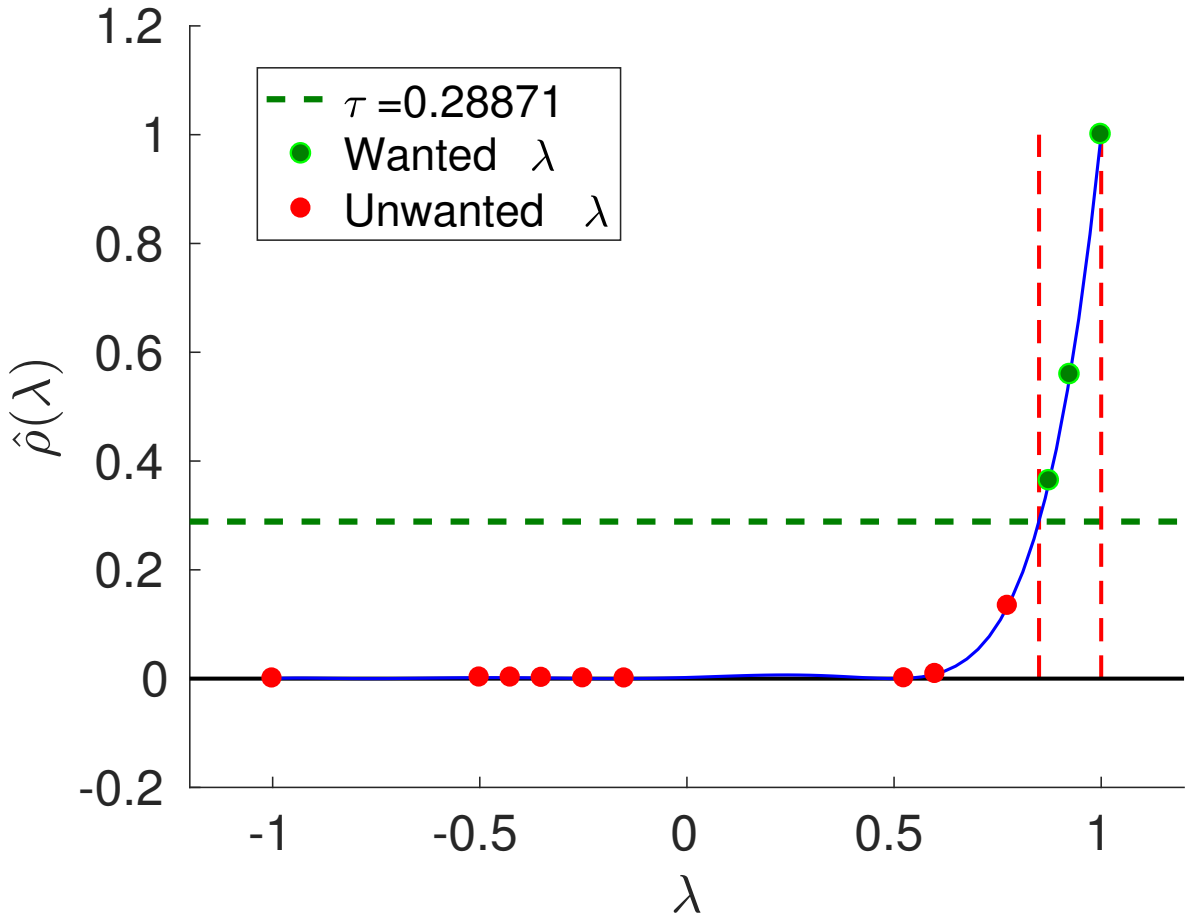

Figure 7: Filter polynomial $\hat{\rho}_k(\lambda)$ (degree = 7) for $s_4 = [7.4, \lambda_{\max}]$

Software and Hardware

- Software: C++11 and *Trilinos*, a C++ software framework for the solution of large-scale, complex multi-physics engineering and scientific problems
- Hardware: EULER cluster
  (*E*rweiterbarer, *U*mweltfreundlicher, *L*eistungsfähiger *E*TH-*R*echner)
  - **Euler I**: 448 nodes; two 12-core Intel Xeon E5-2697v2 per node (2.7 GHz); 64 - 256 GB DDR3 memory
  - **Euler II**: 768 nodes; two 12-core Intel Xeon E5-2697v3 per node (2.5 - 3.3 GHz); 64 - 512 GB of DDR4 memory

Zum Projekt und Masterarbeit allgemein

- Sehr interdisziplinär
  - Entwicklung in C++
  - Deployment auf High-Performance Cluster
  - Einiges an Mathematik
  - Zusammenarbeit mit Physiker am Paul Scherrer Institut (PSI)
- Sehr lehrreiche Zeit
  - Neue Themen
  - Neue Situationen

Fragen

Bei Fragen und weiteren Auskünften zu meinen Erfahrungen stehe ich sehr gerne zur Verfügung, sei es direkt nach dieser Veranstaltung oder auch per Email: g@accaputo.ch

Take care!

References

[1] Ruipeng Li, Yuanzhe Xi, Eugene Vecharynski, Chao Yang, and Yousef Saad. A thick-restart lanczos algorithm with polynomial filtering for hermitian eigenvalue problems. *SIAM Journal on Scientific Computing*, 38(4):A2512–A2534, 2016.

[2] Lin Lin, Yousef Saad, and Chao Yang. Approximating spectral densities of large matrices. *SIAM Review*, 58(1):34–65, 2016.