



## Programmierblatt 1.

Besprechungswoche: 06.10. – 10.10.2025

### Informationen zur Abgabe

Zur Abgabe der Programmieraufgaben findet ein Meeting mit einem Assistenten statt. Ihr Code sollte dann alle Plots auf dem Aufgabenblatt reproduzieren können. Im Meeting müssen Sie Ihren Code erklären und Fragen dazu beantworten können. Die Plots sind immer zu beschriften und mit einer Legende zu versehen, die Ordnung der Konvergenzgerade ist anzugeben.

**Ihr Code muss bis am Sonntag vor der Besprechungswoche um Mitternacht auf ADAM hochgeladen werden.** Beginnen Sie frühzeitig mit der Bearbeitung der Programmieraufgaben. Am Wochenende vor der Abgabe werden keine Fragen mehr beantwortet.

### Zufallsfelder

Gegeben sei ein Gaußsches Zufallsfeld<sup>1</sup>  $a(x, \omega)$  mit  $x \in D$  und  $\omega \in \Omega$ . Zu dessen effizienten numerischen Simulation trennt man die physikalische Variable  $x$  von der stochastischen Variable  $\omega$  mithilfe der Karhunen-Loève-Entwicklung<sup>2</sup> von  $a(x, \omega)$ . Diese liefert die Darstellung

$$a(x, \omega) = \mathbb{E}_a(x) + \sum_{j=1}^{\infty} \sqrt{\lambda_j} \phi_j(x) \psi_j(\omega). \quad (1)$$

Hierbei ist  $\mathbb{E}_a$  das erwartete Zufallsfeld und  $(\lambda_j, \phi_j)$  sind die Eigenpaare des Kovarianzoperators

$$(\mathcal{K}u)(x) = \int_D \kappa(x, y) u(y) dy, \quad (2)$$

$\psi_j$  sind i.i.d. normalverteilte Zufallsvariablen. Die Funktion  $\kappa : D \times D \rightarrow \mathbb{R}$  wird als der Kovarianzkern bezeichnet. Im Folgenden wählen wir der Einfachheit halber  $D = [0, 1]$ .

Ziel dieser Programmieraufgabe ist die numerische Approximation der Karhunen-Loève-Entwicklung. Dazu muss man die Summe in (1) geeignet abschneiden und die Eigenpaare des Kovarianzoperators berechnen. Dies machen wir mit dem sogenannten Nyström-Verfahren, bei welchem die Integration in (2) durch eine Quadraturformel angenähert wird. Speziell für die Mittelpunkregel gilt

$$\int_0^1 \kappa(x, y) u(y) dy \approx \frac{1}{n} \sum_{i=1}^n \kappa(x, y_i^{(n)}) u(y_i^{(n)}), \quad y_i^{(n)} = \frac{i-1}{n} + \frac{1}{2n}. \quad (3)$$

Wir lösen dann das *kontinuierliche* Eigenwertproblem

$$\lambda \phi(x) = \int_D \kappa(x, y) \phi(y) dy$$

mithilfe der Quadraturformel (3) in allen Punkten  $x_i^{(n)} = y_i^{(n)}$  und erhalten das *diskrete* Eigenwertproblem

$$\lambda^{(n)} \phi^{(n)} = \mathbf{K}^{(n)} \phi^{(n)}, \quad \mathbf{K}^{(n)} = \frac{1}{n} \left[ \kappa(x_i^{(n)}, y_j^{(n)}) \right]_{i,j}. \quad (4)$$

<sup>1</sup>Ein Gaußsches Zufallsfeld liegt vor, wenn  $a(x, \omega)$  für jedes  $x \in D$  eine normalverteilte Zufallsgösse ist.

<sup>2</sup>Diese wird oft auch als POD (proper orthogonal decomposition), PCA (principle component analysis) oder SVD (singular value decomposition) bezeichnet

Zuerst möchten wir das diskrete Eigenwertproblem (4) für den Kovarianzkern (und den zugehörigen Operator)

$$\kappa_{\text{exp}}(x, y) = \frac{1}{\sigma} e^{-\frac{|x-y|}{\sigma}}, \quad (\mathcal{K}_{\text{exp}} u)(x) = \int_0^1 \kappa_{\text{exp}}(x, y) u(y) dy. \quad (5)$$

betrachten. Dabei sei die Korrelationslänge  $\sigma = 1$  gewählt.

Es lässt sich zeigen, dass die Eigenpaare zu dem Operator  $\mathcal{K}_{\text{exp}}$  die Form

$$\lambda_\ell = \frac{2}{\gamma_\ell^2 + 1}, \quad \phi_\ell(x) = \frac{1}{\beta_\ell} (\sin(\gamma_\ell x) + \gamma_\ell \cos(\gamma_\ell x))$$

haben. Dabei sind  $\gamma_\ell$  die positiven Lösungen der Gleichung  $(\gamma^2 - 1) \tan(\gamma) - 2\gamma = 0$ . Die  $\beta_\ell$  werden so gewählt, dass  $\|\phi_\ell\|_{L^2(D)} = 1$  gilt. Die Werte  $\gamma_\ell$ ,  $\beta_\ell$  und  $\lambda_\ell$  für die ersten drei Eigenpaare sind in der folgenden Tabelle angegeben:

| $\ell$ | $\gamma_\ell$    | $\beta_\ell$     | $\lambda_\ell$    |
|--------|------------------|------------------|-------------------|
| 1      | 1.30654237418881 | 1.53412075397455 | 0.738810809416452 |
| 2      | 3.67319440630425 | 2.87161602120694 | 0.138003775354263 |
| 3      | 6.58462004256417 | 4.81441694834056 | 0.045088487289781 |

Wir bezeichnen mit  $\phi_\ell^{(n)}$  die durch den  $\ell$ -ten Eigenvektor  $\phi_\ell^{(n)}$  definierte Näherung an  $\phi_\ell(x)$ . Dabei benutzen wir die Konvention, dass die Eigenvektoren normiert sein sollen, d.h.,  $\|\phi^{(n)}\|_2 = 1$ , weshalb die Funktion  $\phi_\ell^{(n)}$  mit dem Faktor  $\sqrt{n}$  gestreckt werden muss. Folglich gilt also  $\phi_\ell^{(n)}(x_i^{(n)}) = \sqrt{n}[\phi_\ell^{(n)}]_i$ .

**Aufgabe 1.** Nutzen Sie die Matlab-Funktion `eig`, um für  $n = 2^7$  die Eigenpaare der Matrix  $\mathbf{K}_{\text{exp}}^{(n)}$  auszurechnen. Stellen Sie die ersten drei dominanten Eigenfunktionen, d.h. Eigenfunktionen zu den drei betragsgrössten Eigenwerten, graphisch dar. Beachten Sie, dass die Funktion `eig` die berechneten Eigenwerte nicht sortiert.

**Aufgabe 2.** Untersuchen Sie die Konvergenz der ersten drei Eigenpaare in Abhängigkeit von  $n$ . Betrachten Sie dazu  $n = 2^k$ ,  $k = 6, 7, \dots, 11$ . Stellen Sie in einem *loglog*-Plot  $|\lambda_\ell - \lambda_\ell^{(n)}|$ , und im gleichen Plot  $\|\phi_\ell - \phi_\ell^{(n)}\|$  in Abhängigkeit von  $n$  dar, wobei  $\phi_\ell := [\phi_\ell(x_i)]_i$  bezeichne. Diskutieren Sie die Konvergenzraten für die Eigenwerte und die Eigenvektoren.

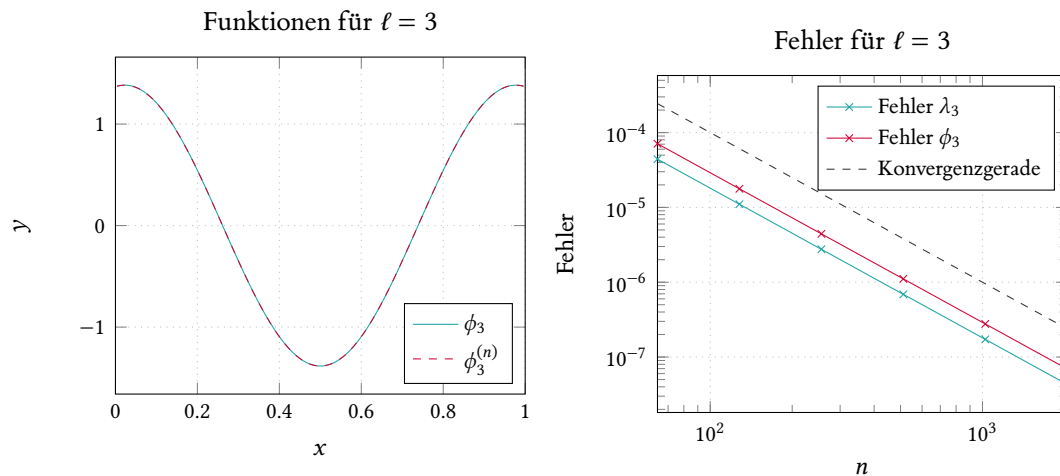
*Hinweis. Beachten Sie, dass die Vektoren  $\phi_\ell$  und  $\phi_\ell^{(n)}$  unterschiedliche Vorzeichen haben können.*

### Niedrigrangapproximation

Zur Lösung des Eigenwertproblems  $\lambda \mathbf{u} = \mathbf{K} \mathbf{u}$  aus (4) wollen wir die *pivotisierte Cholesky-Zerlegung*  $\mathbf{K} \approx \mathbf{L} \mathbf{L}^\top$  verwenden, welche gemäss dem unten stehenden Algorithmus bestimmt wird. Beachten Sie, dass die Matrix  $\mathbf{K}$  für den Algorithmus nicht explizit bekannt sein muss. Es genügt, eine Routine bereitzustellen, die jeweils die aktuell benötigten Matrixeinträge berechnet. Insbesondere bei grossen Matrizen reduziert sich dadurch der Speicher- und Rechenaufwand erheblich.

**Aufgabe 3.** Implementieren Sie eine Matlab-Funktion

```
function [L, p] = pivChol(k, xs, tol),
```




---

#### Algorithmus pivotisierte Cholesky-Zerlegung

---

*Input:* Kernfunktion  $k : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ , Punkte  $\mathbf{x}_1, \dots, \mathbf{x}_n$  Genauigkeit  $\text{tol} \geq 0$ .

*Output:*  $\mathbf{L} \in \mathbb{R}^{n \times m}$  mit  $\mathbf{L}\mathbf{L}^\top \approx \mathbf{K}$ , Permutationsvektor  $\mathbf{p}$ .

---

- 1: setze  $\mathbf{p} := [1, 2, \dots, n]$ ,  $\mathbf{d} := [k(\mathbf{x}_j, \mathbf{x}_j)]_{j=1}^n$ ,  $\varepsilon := \|\mathbf{d}\|_1$ ,  $m := 1$
  - 2: **while**  $m \leq n$  and  $\varepsilon \geq \text{tol}$  **do**
  - 3:   setze  $\text{pivot} := \operatorname{argmax}_{\ell \in \{m, m+1, \dots, n\}} d_{p_\ell}$
  - 4:   vertausche  $p_{\text{pivot}} \leftrightarrow p_m$
  - 5:   setze  $\mathbf{L}_{p_m, m} := \sqrt{d_{p_m}}$
  - 6:   setze  $\mathbf{L}_{p_{m+1:n}, m} := [k(\mathbf{x}_{p_{m+1}}, \mathbf{x}_{p_m}), \dots, k(\mathbf{x}_{p_n}, \mathbf{x}_{p_m})]^\top / \mathbf{L}_{p_m, m}$
  - 7:   update  $\mathbf{L}_{p_{m+1:n}, m} := \mathbf{L}_{p_{m+1:n}, m} - \mathbf{L}_{p_{m+1:n}, 1:m-1} \cdot \mathbf{L}_{p_m, 1:m-1}^\top / \mathbf{L}_{p_m, m}$
  - 8:   update  $\mathbf{d}_{p_{m:n}} := \mathbf{d}_{p_{m:n}} - [\mathbf{L}_{p_m, m}^2, \mathbf{L}_{p_{m+1}, m}^2, \dots, \mathbf{L}_{p_n, m}^2]^\top$
  - 9:   update  $\varepsilon := \|\mathbf{d}_{p_{m:n}}\|_1$
  - 10:   setze  $m := m + 1$
  - 11: **end while**
- 

welche die pivotisierte Cholesky-Zerlegung berechnet. Das Argument  $k$  sei hierbei ein function handle, das eine Kernfunktion  $k(x, y)$  vektorisiert auswertet. Beachten Sie, dass die Matrix  $\mathbf{K}$  in dem Algorithmus nie explizit aufgestellt wird. Der Parameter  $n$  entspricht der Dimension der Matrix  $\mathbf{K}$  und  $\text{tol}$  gibt die Abbruchgenauigkeit vor. Es sollen abgesehen von der while-Schleife keine weiteren Schleifen verwendet werden.

Testen Sie Ihre Implementierung mit dem folgenden Code, das Resultat soll dabei von der Größenordnung  $10^{-10}$  sein:

---

```
k = @(x, y) exp(-(x-y).^2);
xs = linspace(0, 1, 4001);
A = k(xs, xs. ');
[L, p] = pivChol(k, xs, 1e-9);
disp(norm(A - L*L.', "fro"));
```

---

#### Aufgabe 4. Betrachten Sie den Gaußschen Kovarianzkern

$$\kappa_{\text{Gauß}}(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{|x-y|^2}{2\sigma^2}}$$

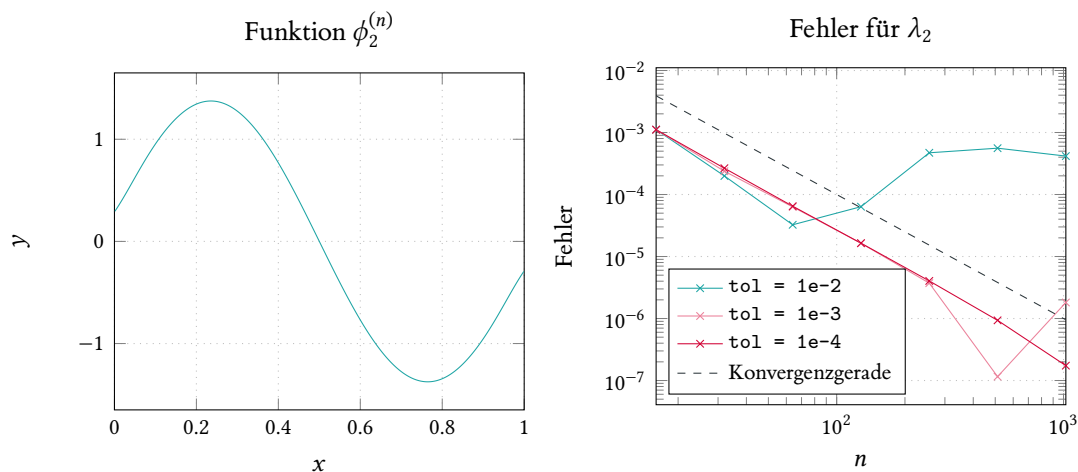
mit  $\sigma = 0.05$ . Berechnen Sie die Eigenpaare der Matrix  $\mathbf{K}_{\text{Gauß}}^{(n)}$ , definiert durch den Kern  $\kappa_{\text{Gauß}}$ , mithilfe der Niedrigrangapproximation. Wählen Sie die Parameter  $\text{tol} = 10^{-8}$  und  $n = 2^8$ .

Sind nun  $\{(\lambda_\ell, \tilde{\Phi}_\ell^{(n)})\}$  die Eigenpaare von  $L^T L$ , so sind

$$\{(\lambda_\ell, \Phi_\ell^{(n)})\}, \quad \Phi_\ell^{(n)} := \frac{1}{\sqrt{\lambda_\ell}} L \tilde{\Phi}_\ell^{(n)}$$

Eigenpaare der grossen Matrix  $LL^T$ . Bestimmen Sie die Eigenwerte und Eigenvektoren  $\tilde{\Phi}_\ell^{(n)}$  des kleinen Eigenwertproblems für  $L^T L$  mit der Matlab-Funktion `eig` und stellen Sie für  $\ell = 1, 2, 3$  die zu  $\Phi_\ell^{(n)}$  gehörigen Funktionen  $\phi_\ell^{(n)}$  graphisch dar.

**Aufgabe 5.** Untersuchen Sie für jede der Abbruchgenauigkeiten  $\text{tol} = 10^{-m}$ ,  $m = 2, 3, 4$ , die Konvergenz der ersten drei Eigenwerte in Abhängigkeit von  $n$ . Betrachten Sie dazu jeweils  $n = 2^\ell$ ,  $\ell = 4, 5, \dots, 10$ . Nehmen Sie als Referenzen die von der Funktion `eig` berechneten Eigenwerte der Matrix  $K_{\text{Gauß}}^{(n)}$  für  $n = 2^{11}$ . Stellen Sie in einem *loglog*-Plot die Grössen  $|\lambda_\ell - \lambda_\ell^{(n)}|$  in Abhängigkeit von  $n$  für die verschiedenen Werte von `tol` dar. Diskutieren Sie die Konvergenz der Eigenwerte.



## Rayleigh-Quotient-Iteration

Als nächstes möchten wir die Rayleigh-Quotient-Iteration implementieren. Dazu betrachten wir den Exponentialkern (5) mit  $\sigma = 1000$ , die Matrix  $K_{\text{exp}}^{(n)}$  sei entsprechend zu (4) definiert. Wir nutzen  $\text{tol} = 10^{-8}$  für die Abbruchgenauigkeit der pivotisierten Cholesky-Zerlegung, sowie  $n = 2^{12}$  für die Quadratur.

**Aufgabe 6.** Schreiben Sie eine Funktion

```
function [mus, zs] = rayleighQuotient(A, mu0, z0, maxIter),
```

welche die Rayleigh-Quotient-Iteration aus der Vorlesung implementiert. Hierbei seien  $A \in \mathbb{R}^{n \times n}$  eine symmetrische und positiv definite Matrix, sowie  $(\mu_0, z_0) \in \mathbb{R} \times \mathbb{R}^n$  eine Startnäherung. Der Rückgabewert `mus` soll dabei ein Zeilenvektor sein, `zs` eine Matrix, welche in der  $k$ -ten Spalte die Annäherung an den Eigenvektor im  $k$ -ten Schritt enthalte. Brechen Sie die Iteration nach dem  $k$ -ten Schritt ab, falls  $k > \text{maxIter}$  oder  $\|\tilde{z}_k\|_2 > 10^{14}$  gilt.

**Aufgabe 7.** Bestimmen Sie mithilfe der Rayleigh-Iteration eine Annäherung an das dominante Eigenpaar von  $K_{\text{exp}}^{(n)}$ , indem Sie zuerst eine Niedrigrangapproximation  $K_{\text{exp}}^{(n)} \approx LL^T$  berechnen

und danach das dominante Eigenpaar der kleineren Matrix  $\mathbf{L}^T \mathbf{L}$  berechnen. Wählen Sie dabei als Startnäherungen  $\mu_0 = 1$  und  $\mathbf{z}_0 = \mathbf{e}_1$ .

Stellen Sie die Fehler  $|\mu_k - \lambda_1|$  und  $\|\mathbf{z}_k - \mathbf{v}_1\|_2$  in der  $k$ -ten Iteration im Vergleich zu den wahren Eigenwerten in einem *semilogy*-Plot graphisch dar. Sie sollte dabei exponentielle Konvergenz beobachten.

Hinweis. Die wahren Werte für  $\lambda_1$  und  $\mathbf{v}_1$  (der Matrix  $\mathbf{L}^T \mathbf{L}$ ) sollen dabei mit der Funktion `eig` bestimmt werden.

### Simulation Gaußscher Zufallsfelder

Mit den nun implementierten Funktionen können wir ein Gaußsches Zufallsfeld, welches durch (1) und einen Kovarianzkern gegeben ist, effizient modellieren. Da es in der Praxis zu aufwändig ist, die exakten Eigenwerte und Eigenvektoren der grossen Matrix  $\mathbf{K}^{(n)}$  zu berechnen, wird in einem ersten Schritt die Matrix  $\mathbf{K}^{(n)}$  durch eine Niedrigrangapproximation  $\mathbf{L}\mathbf{L}^T$  angenähert. In einem zweiten Schritt werden dann die Eigenpaare  $(\lambda_j, \tilde{\phi}_j^{(n)})$  der kleinen Matrix  $\mathbf{L}^T \mathbf{L}$  berechnet. Folglich liefert für i.i.d. Normalverteilte  $\psi_j$  die Entwicklung

$$\mathbf{X}(\omega) := \sqrt{n} \sum_{j=1}^k \mathbf{L} \tilde{\phi}_j^{(n)} \psi_j(\omega) \quad (6)$$

eine zentrierte Zufallsgrösse, welche die Punktauswertungen von  $a(\cdot, \omega)$  in den Punkten  $x_i$  enthält.

**Aufgabe 8.** Schreiben Sie ein Skript, welches sowohl für den Gauss-Kern, als auch den Exponentialkern eine Realisierung des Zufallsfeldes  $a$  gemäss (6) berechnet. Standardnormalverteilte Variablen können dabei mit dem Befehl `randn` generiert werden. Benutzen Sie dafür  $n = 2^9$ ,  $\sigma = 0.05$  und `tol` =  $10^{-9}$ . Wählen Sie dabei  $k$  als die Anzahl Spalten in  $\mathbf{L}$ . Was fällt dabei auf? Wie verhält sich  $k$ ?

