Black Box Probabilistic Numerics

Chris. J. Oates

April 2024
Probabilistic Numerics Spring School

Motivation

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$U|(A = a) \sim \mathcal{GP}(m_{U|a}, k_{U|a})$$
$$m_{U|a}(t) = A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a$$
$$k_{U|a}(t, t') = k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t')$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$U|(A = a) \sim \mathcal{GP}(m_{U|a}, k_{U|a})$$
$$m_{U|a}(t) = A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a$$
$$k_{U|a}(t, t') = k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t')$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

Bayesian Probabilistic Numerical Methods: Conjugate Setting

- **State of the universe**: $u = (u(t))_{t \in T}$, $u \in \mathcal{U}$
- **Information**: $A : \mathcal{U} \to \mathbb{R}^n$, some $n \in \{1, 2, \dots\}$
- **Quantity of interest**: $Q : \mathcal{U} \to \mathbb{R}^m$, some $m \in \{1, 2, \dots\} \cup \{\infty\}$

e.g. for numerical integration we might have

$$Q(u) = \int_0^1 u(t) \, \mathrm{d}t, \qquad A(u) = [u(0), u(x), u(2x), \dots, u(1)] \, .$$

Linear information enables us to use a conjugate Gaussian framework:

1. Select a Gaussian process $(U(t))_{t \in T}$ to represent epistemic uncertainty in $(u(t))_{t \in T}$.
2. Compute the conditional

$$\begin{aligned} U|(A = a) &\sim \mathcal{GP}(m_{U|a}, k_{U|a}) \\ m_{U|a}(t) &= A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} a \\ k_{U|a}(t, t') &= k(t, t') - A_{t'} k(t, t')[A_t A_{t'} k(t, t')]^{-1} A_t k(t, t') \end{aligned}$$

3. Push the remaining uncertainty through Q.

What About Nonlinear Information?

Using the same notation, consider instead

$$Mu = b, \qquad u = (u_1, \ldots, u_d)^\top \in \mathbb{R}^d.$$

The matrix-vector products computed in the popular conjugate gradient method are

$$\begin{aligned}
&\langle s^{(1)}, b\rangle, && s^{(1)} = b \\
&\langle s^{(2)}, b\rangle, && s^{(2)} = \text{cubic in } b \\
&\langle s^{(3)}, b\rangle, && s^{(3)} = \text{ninth powers of } b \\
&\quad\vdots && \quad\vdots
\end{aligned}$$

So it seems natural to let

$$A(u) = \begin{bmatrix} \langle s^{(1)}, Mu\rangle \\ \langle s^{(2)}, Mu\rangle \\ \vdots \end{bmatrix}$$

... but this is **nonlinear** information!

This problem is not easily fixed.

What About Nonlinear Information?

Using the same notation, consider instead

$$Mu = b, \qquad u = (u_1, \ldots, u_d)^\top \in \mathbb{R}^d.$$

The matrix-vector products computed in the popular conjugate gradient method are

$$\begin{aligned} &\langle s^{(1)}, b\rangle, && s^{(1)} = b \\ &\langle s^{(2)}, b\rangle, && s^{(2)} = \text{cubic in } b \\ &\langle s^{(3)}, b\rangle, && s^{(3)} = \text{ninth powers of } b \\ &\quad\vdots && \quad\vdots \end{aligned}$$

So it seems natural to let

$$A(u) = \begin{bmatrix} \langle s^{(1)}, Mu\rangle \\ \langle s^{(2)}, Mu\rangle \\ \vdots \end{bmatrix}$$

... but this is **nonlinear** information!

This problem is not easily fixed.

What About Nonlinear Information?

Using the same notation, consider instead

$$Mu = b, \qquad u = (u_1, \ldots, u_d)^\top \in \mathbb{R}^d.$$

The matrix-vector products computed in the popular conjugate gradient method are

$$\begin{aligned} &\langle s^{(1)}, Mu \rangle, && s^{(1)} = b \\ &\langle s^{(2)}, Mu \rangle, && s^{(2)} = \text{cubic in } b \\ &\langle s^{(3)}, Mu \rangle, && s^{(3)} = \text{ninth powers of } b \\ &\vdots && \vdots \end{aligned}$$

So it seems natural to let

$$A(u) = \begin{bmatrix} \langle s^{(1)}, Mu \rangle \\ \langle s^{(2)}, Mu \rangle \\ \vdots \end{bmatrix}$$

... but this is **nonlinear** information!

This problem is not easily fixed.

What About Nonlinear Information?

Using the same notation, consider instead

$$Mu = b, \qquad u = (u_1, \ldots, u_d)^\top \in \mathbb{R}^d.$$

The matrix-vector products computed in the popular conjugate gradient method are

$$\begin{aligned} &\langle s^{(1)}, Mu \rangle, && s^{(1)} = b \\ &\langle s^{(2)}, Mu \rangle, && s^{(2)} = \text{cubic in } b \\ &\langle s^{(3)}, Mu \rangle, && s^{(3)} = \text{ninth powers of } b \\ &\quad \vdots && \quad \vdots \end{aligned}$$

So it seems natural to let

$$A(u) = \begin{bmatrix} \langle s^{(1)}, Mu \rangle \\ \langle s^{(2)}, Mu \rangle \\ \vdots \end{bmatrix}$$

... but this is **nonlinear** information!

This problem is not easily fixed.

What About Nonlinear Information?

Using the same notation, consider instead

$$Mu = b, \qquad u = (u_1, \ldots, u_d)^\top \in \mathbb{R}^d.$$

The matrix-vector products computed in the popular conjugate gradient method are

$$\begin{aligned} &\langle s^{(1)}, Mu \rangle, && s^{(1)} = Mu \\ &\langle s^{(2)}, Mu \rangle, && s^{(2)} = \text{cubic in } u \\ &\langle s^{(3)}, Mu \rangle, && s^{(3)} = \text{ninth powers of } u \\ &\quad \vdots && \quad \vdots \end{aligned}$$

So it seems natural to let

$$A(u) = \begin{bmatrix} \langle s^{(1)}, Mu \rangle \\ \langle s^{(2)}, Mu \rangle \\ \vdots \end{bmatrix}$$

... but this is **nonlinear** information!

This problem is not easily fixed.

What About Nonlinear Information?

Using the same notation, consider instead

$$Mu = b, \qquad u = (u_1, \ldots, u_d)^\top \in \mathbb{R}^d.$$

The matrix-vector products computed in the popular conjugate gradient method are

$$\begin{aligned} &\langle s^{(1)}, Mu \rangle, && s^{(1)} = Mu \\ &\langle s^{(2)}, Mu \rangle, && s^{(2)} = \text{cubic in } u \\ &\langle s^{(3)}, Mu \rangle, && s^{(3)} = \text{ninth powers of } u \\ &\quad \vdots && \quad \vdots \end{aligned}$$

So it seems natural to let

$$A(u) = \begin{bmatrix} \langle s^{(1)}, Mu \rangle \\ \langle s^{(2)}, Mu \rangle \\ \vdots \end{bmatrix}$$

... but this is **nonlinear** information!

This problem is not easily fixed.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of probabilistic numerics (PN).

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

- (✓) applicable to nonlinear information
- (✓) state-of-the-art performance and functionality (in principle, at least)
- (✓) provably higher order of convergence relative to a single application of the numerical method
- (✗) multiple realisations of a numerical method are required
- (✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Aim of the Talk

Aim: A *black box* that enables state-of-the-art numerical algorithms to be immediately exploited in the context of PN.

Key Idea: Predict the limit of a sequence of increasingly accurate approximations produced by a traditional numerical method.

Bonus: A statistical perspective on *extrapolation methods*.

GPs: For concreteness, we will predict using GPs, but other predictive models could be used.

Compared to standard PN:

(✓) applicable to nonlinear information

(✓) state-of-the-art performance and functionality (in principle, at least)

(✓) provably higher order of convergence relative to a single application of the numerical method

(✗) multiple realisations of a numerical method are required

(✗) a joint statistical model has to be built for not just the quantity of interest but also for the error associated with the output of a traditional numerical method.

Extrapolation Methods

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \boldsymbol{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \boldsymbol{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \boldsymbol{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \mathbf{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \mathbf{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- modern computer codes

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \boldsymbol{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \mathbf{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \mathbf{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

Extrapolation methods

An *extrapolation method* is an estimate for the limit

$$\lim_{\boldsymbol{x} \to \mathbf{0}} f(\boldsymbol{x})$$

which can be based on values $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ with $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset (0, \infty)^d$ such that the associated computational cost falls within a notional budget.

Applications:

- finite difference approximation to derivatives
- numerical integration
- numerical solution of differential equations
- **modern computer codes**

Solutions:

- Richardson [1911] (higher-order convergence guaranteed)
- other extrapolation methods (NA-informed, typically univariate quantity of interest)
- multi-fidelity modelling (flexible, data-driven, multivariate quantity of interest)

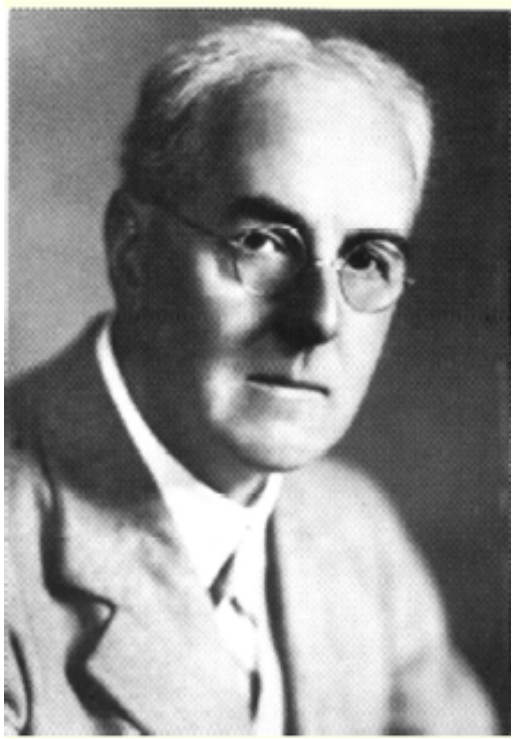


Figure: Lewis Fry Richardson (1881 - 1953; born in Newcastle upon Tyne)

Example: Higher-order convergence with Richardson extrapolation

Suppose that

$$\underbrace{f(x)}_{\text{numerical method}} = \underbrace{f(0)}_{\text{quantity of interest}} + \underbrace{f'(0)x + O(x^2)}_{\text{error of the numerical method}}$$

so that $f(x)$ is a *first-order* approximation to $f(0)$.

Then

$$\begin{aligned} & 2f(x) - f(2x) \\ &= 2[f(0) + f'(0)x + O(x^2)] \\ &\quad - [f(0) + f'(0)(2x) + O(x^2)] \\ &= f(0) + O(x^2) \end{aligned}$$

is a *second-order* approximation to $f(0)$.

In general

$$f^{(s)} \text{ cts} \implies \text{combine } s \text{ evaluations of } f \text{ to get order-}s \text{ approximation to } f(0) \ .$$

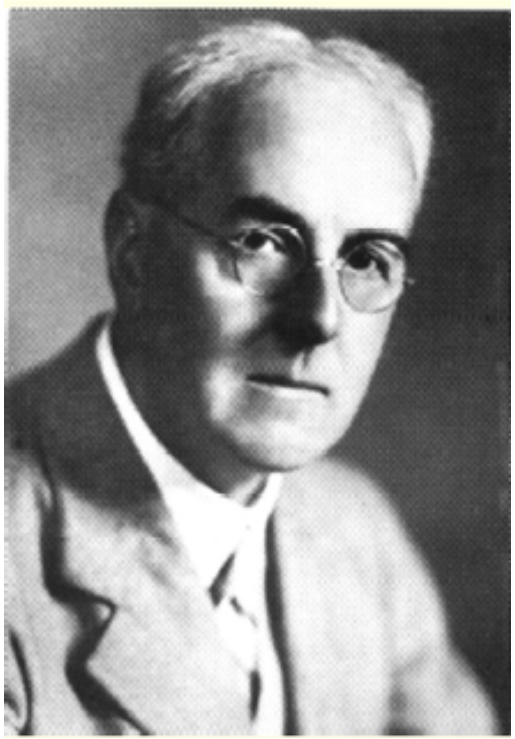


Figure: Lewis Fry Richardson (1881 - 1953; born in Newcastle upon Tyne)

Example: Higher-order convergence with Richardson extrapolation

Suppose that

$$\underbrace{f(x)}_{\text{numerical method}} = \underbrace{f(0)}_{\text{quantity of interest}} + \underbrace{f'(0)x + O(x^2)}_{\text{error of the numerical method}}$$

so that $f(x)$ is a *first-order* approximation to $f(0)$.

Then

$$\begin{aligned} 2f(x) - f(2x) &= 2[f(0) + f'(0)x + O(x^2)] \\ &\quad - [f(0) + f'(0)(2x) + O(x^2)] \\ &= f(0) + O(x^2) \end{aligned}$$

is a *second-order* approximation to $f(0)$.

In general

$f^{(s)}$ cts $\implies$ combine s evaluations of f to get order-s approximation to $f(0)$.

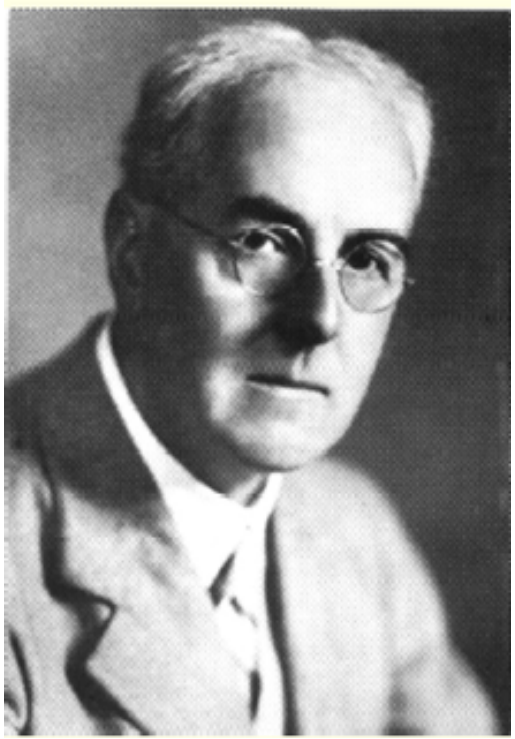


Figure: Lewis Fry Richardson (1881 - 1953; born in Newcastle upon Tyne)

Example: Higher-order convergence with Richardson extrapolation

Suppose that

$$\underbrace{f(x)}_{\text{numerical method}} = \underbrace{f(0)}_{\text{quantity of interest}} + \underbrace{f'(0)x + O(x^2)}_{\text{error of the numerical method}}$$

so that $f(x)$ is a *first-order* approximation to $f(0)$.

Then

$$\begin{aligned} 2f(x) - f(2x) &= 2[f(0) + f'(0)x + O(x^2)] \\ &\quad - [f(0) + f'(0)(2x) + O(x^2)] \\ &= f(0) + O(x^2) \end{aligned}$$

is a *second-order* approximation to $f(0)$.

In general

$$f^{(s)} \text{ cts} \implies \text{combine } s \text{ evaluations of } f \text{ to get order-}s \text{ approximation to } f(0) \ .$$

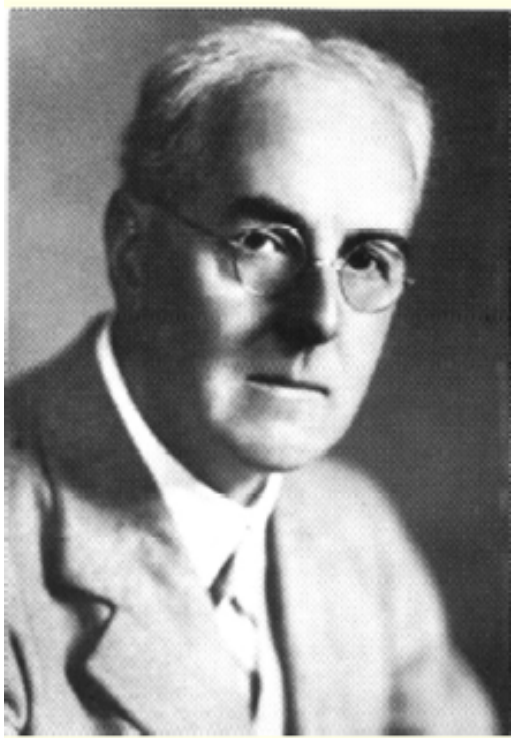


Figure: Lewis Fry Richardson (1881 - 1953; born in Newcastle upon Tyne)

Example: Higher-order convergence with Richardson extrapolation

Suppose that

$$\underbrace{f(x)}_{\text{numerical method}} = \underbrace{f(0)}_{\text{quantity of interest}} + \underbrace{f'(0)x + O(x^2)}_{\text{error of the numerical method}}$$

so that $f(x)$ is a *first-order* approximation to $f(0)$.

Then

$$\begin{aligned} 2f(x) - f(2x) &= 2[f(0) + f'(0)x + O(x^2)] \\ &\quad - [f(0) + f'(0)(2x) + O(x^2)] \\ &= f(0) + O(x^2) \end{aligned}$$

is a *second-order* approximation to $f(0)$.

In general

$f^{(s)}$ cts $\implies$ combine s evaluations of f to get order-s approximation to $f(0)$.

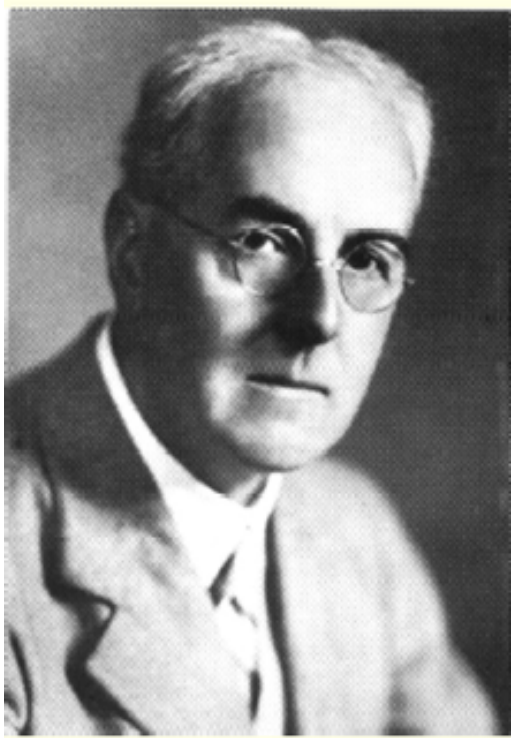


Figure: Lewis Fry Richardson (1881 - 1953; born in Newcastle upon Tyne)

Example: Higher-order convergence with Richardson extrapolation

Suppose that

$$\underbrace{f(x)}_{\text{numerical method}} = \underbrace{f(0)}_{\text{quantity of interest}} + \underbrace{f'(0)x + O(x^2)}_{\text{error of the numerical method}}$$

so that $f(x)$ is a *first-order* approximation to $f(0)$.

Then

$$\begin{aligned} 2f(x) - f(2x) \\ &= 2[f(0) + f'(0)x + O(x^2)] \\ &\quad - [f(0) + f'(0)(2x) + O(x^2)] \\ &= f(0) + O(x^2) \end{aligned}$$

is a *second-order* approximation to $f(0)$.

In general

$f^{(s)}$ cts $\implies$ combine s evaluations of f to get order-s approximation to $f(0)$.

Figure: Christiaan Huygens (1629 -1695; born in the Hague)

Example: Higher-order convergence with Richardson extrapolation

Suppose that

$$\underbrace{f(x)}_{\text{numerical method}} = \underbrace{f(0)}_{\text{quantity of interest}} + \underbrace{f'(0)x + O(x^2)}_{\text{error of the numerical method}}$$

so that $f(x)$ is a *first-order* approximation to $f(0)$.

Then

$$\begin{aligned} 2f(x) - f(2x) \\ &= 2[f(0) + f'(0)x + O(x^2)] \\ &\quad - [f(0) + f'(0)(2x) + O(x^2)] \\ &= f(0) + O(x^2) \end{aligned}$$

is a *second-order* approximation to $f(0)$.

In general

$f^{(s)}$ cts $\implies$ combine s evaluations of f to get order-s approximation to $f(0)$.

Problems:

- extrapolation methods do not provide probabilistic UQ
- multi-fidelity modelling is not NA-informed and lacks higher-order convergence guarantees
- neither method enables **experimental design** for $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

Seek a extrapolation method that:

- provides probabilistic UQ
- is NA-informed
- applicable to scenarios where multi-fidelity modelling used.

Problems:

- extrapolation methods do not provide probabilistic UQ
- multi-fidelity modelling is not NA-informed and lacks higher-order convergence guarantees
- neither method enables **experimental design** for $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

Seek a extrapolation method that:

- provides probabilistic UQ
- is NA-informed
- applicable to scenarios where multi-fidelity modelling used.

Problems:

- extrapolation methods do not provide probabilistic UQ
- multi-fidelity modelling is not NA-informed and lacks higher-order convergence guarantees
- neither method enables **experimental design** for $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

Seek a extrapolation method that:

- provides probabilistic UQ
- is NA-informed
- applicable to scenarios where multi-fidelity modelling used.

Problems:

- extrapolation methods do not provide probabilistic UQ
- multi-fidelity modelling is not NA-informed and lacks higher-order convergence guarantees
- neither method enables **experimental design** for $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

Seek a extrapolation method that:

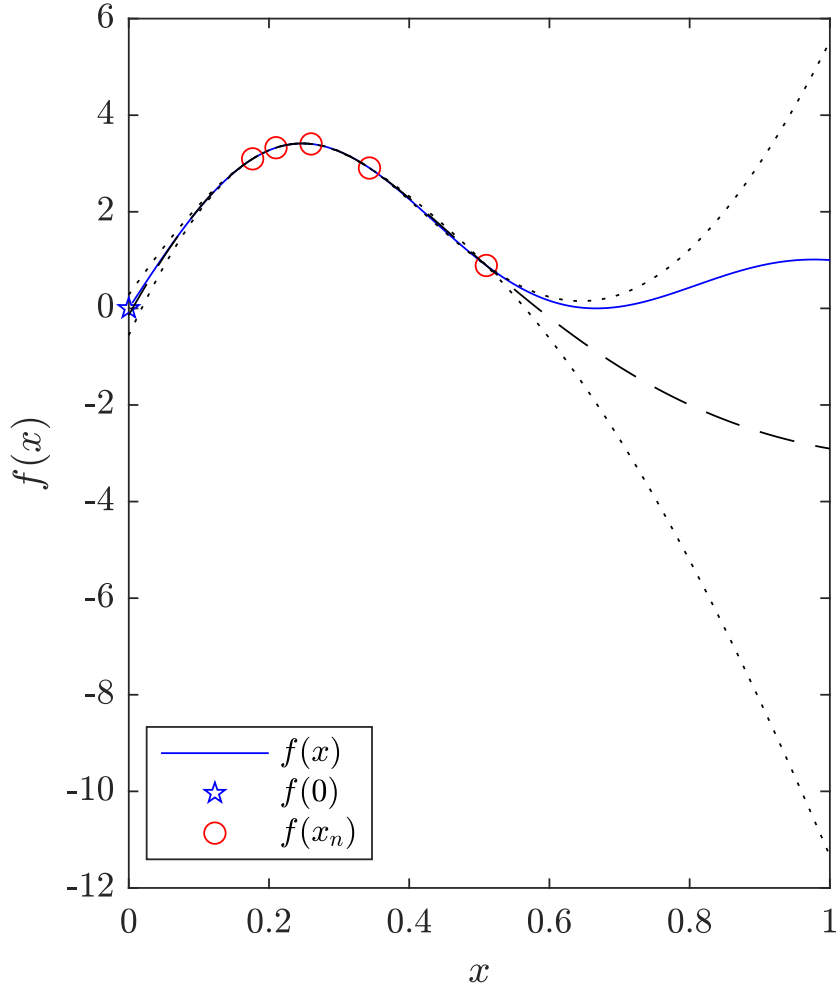
- provides probabilistic UQ
- is NA-informed
- applicable to scenarios where multi-fidelity modelling used.

Problems:

- extrapolation methods do not provide probabilistic UQ
- multi-fidelity modelling is not NA-informed and lacks higher-order convergence guarantees
- neither method enables **experimental design** for $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

Seek a extrapolation method that:

- provides probabilistic UQ
- is NA-informed
- applicable to scenarios where multi-fidelity modelling used.



Black Box Probabilistic Numerics

Given an explicit error bound b such that $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ and $b(\mathbf{0}) = 0$.

How to encode this knowledge into a probabilistic regression model?

Gauss–Richardson Extrapolation (GRE)

Samples $g \sim \mathcal{GP}(0, k)$ from a Gaussian process with covariance kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \left\{ k_0^2 + b(\mathbf{x}) b(\mathbf{x}') k_e(\mathbf{x}, \mathbf{x}') \right\}, \qquad \mathbf{x}, \mathbf{x}' \in (0, \infty)^d,$$

a.s. satisfy $g(\mathbf{x}) - g(\mathbf{0}) = O(b(\mathbf{x}))$ as well.

So we use this GP regression model, trained on $[f(X_n)]_i = f(\mathbf{x}_i)$, to predict $f(\mathbf{0})$.

$\implies$ Only need to work with *linear* information!

Since k_0^2 is proportional to the prior variance for $f(\mathbf{0})$, we seek to let $k_0^2 \to \infty$, representing the flat prior limit / universal kriging:

$$f(\mathbf{0}) | f(X_n) \sim \mathcal{N}(m_n[f], v_n[f]), \qquad m_n[f] = \frac{\mathbf{1}^\top \mathbf{K}_b^{-1} f(X_n)}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}}, \qquad v_n[f] = \frac{\sigma_n^2[f]}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}},$$

where $[\mathbf{K}_b]_{i,j} = k_b(\mathbf{x}_i, \mathbf{x}_j) = b(\mathbf{x}_i) b(\mathbf{x}_j) k_e(\mathbf{x}_i, \mathbf{x}_j)$ and $\sigma_n^2[f]$ is a scale estimator, to be specified.

Given an explicit error bound b such that $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ and $b(\mathbf{0}) = 0$.

How to encode this knowledge into a probabilistic regression model?

Gauss–Richardson Extrapolation (GRE)

Samples $g \sim \mathcal{GP}(0, k)$ from a Gaussian process with covariance kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \left\{ k_0^2 + b(\mathbf{x}) b(\mathbf{x}') k_e(\mathbf{x}, \mathbf{x}') \right\}, \qquad \mathbf{x}, \mathbf{x}' \in (0, \infty)^d,$$

a.s. satisfy $g(\mathbf{x}) - g(\mathbf{0}) = O(b(\mathbf{x}))$ as well.

So we use this GP regression model, trained on $[f(X_n)]_i = f(\mathbf{x}_i)$, to predict $f(\mathbf{0})$.

$\implies$ Only need to work with *linear* information!

Since k_0^2 is proportional to the prior variance for $f(\mathbf{0})$, we seek to let $k_0^2 \to \infty$, representing the flat prior limit / universal kriging:

$$f(\mathbf{0}) | f(X_n) \sim \mathcal{N}(m_n[f], v_n[f]), \qquad m_n[f] = \frac{\mathbf{1}^\top \mathbf{K}_b^{-1} f(X_n)}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}}, \qquad v_n[f] = \frac{\sigma_n^2[f]}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}},$$

where $[\mathbf{K}_b]_{i,j} = k_b(\mathbf{x}_i, \mathbf{x}_j) = b(\mathbf{x}_i) b(\mathbf{x}_j) k_e(\mathbf{x}_i, \mathbf{x}_j)$ and $\sigma_n^2[f]$ is a scale estimator, to be specified.

Given an explicit error bound b such that $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ and $b(\mathbf{0}) = 0$.

How to encode this knowledge into a probabilistic regression model?

Gauss–Richardson Extrapolation (GRE)

Samples $g \sim \mathcal{GP}(0, k)$ from a Gaussian process with covariance kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \left\{ k_0^2 + b(\mathbf{x}) b(\mathbf{x}') k_e(\mathbf{x}, \mathbf{x}') \right\}, \qquad \mathbf{x}, \mathbf{x}' \in (0, \infty)^d,$$

a.s. satisfy $g(\mathbf{x}) - g(\mathbf{0}) = O(b(\mathbf{x}))$ as well.

So we use this GP regression model, trained on $[f(X_n)]_i = f(\mathbf{x}_i)$, to predict $f(\mathbf{0})$.

$\implies$ Only need to work with *linear* information!

Since k_0^2 is proportional to the prior variance for $f(\mathbf{0})$, we seek to let $k_0^2 \to \infty$, representing the flat prior limit / universal kriging:

$$f(\mathbf{0}) | f(X_n) \sim \mathcal{N}(m_n[f], v_n[f]), \qquad m_n[f] = \frac{\mathbf{1}^\top \mathbf{K}_b^{-1} f(X_n)}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}}, \qquad v_n[f] = \frac{\sigma_n^2[f]}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}},$$

where $[\mathbf{K}_b]_{i,j} = k_b(\mathbf{x}_i, \mathbf{x}_j) = b(\mathbf{x}_i) b(\mathbf{x}_j) k_e(\mathbf{x}_i, \mathbf{x}_j)$ and $\sigma_n^2[f]$ is a scale estimator, to be specified.

Given an explicit error bound b such that $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ and $b(\mathbf{0}) = 0$.

How to encode this knowledge into a probabilistic regression model?

Gauss–Richardson Extrapolation (GRE)

Samples $g \sim \mathcal{GP}(0, k)$ from a Gaussian process with covariance kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \left\{ k_0^2 + b(\mathbf{x}) b(\mathbf{x}') k_e(\mathbf{x}, \mathbf{x}') \right\}, \qquad \mathbf{x}, \mathbf{x}' \in (0, \infty)^d,$$

a.s. satisfy $g(\mathbf{x}) - g(\mathbf{0}) = O(b(\mathbf{x}))$ as well.

So we use this GP regression model, trained on $[f(X_n)]_i = f(\mathbf{x}_i)$, to predict $f(\mathbf{0})$.

$\implies$ Only need to work with *linear* information!

Since k_0^2 is proportional to the prior variance for $f(\mathbf{0})$, we seek to let $k_0^2 \to \infty$, representing the flat prior limit / universal kriging:

$$f(\mathbf{0}) | f(X_n) \sim \mathcal{N}(m_n[f], v_n[f]), \qquad m_n[f] = \frac{\mathbf{1}^\top \mathbf{K}_b^{-1} f(X_n)}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}}, \qquad v_n[f] = \frac{\sigma_n^2[f]}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}},$$

where $[\mathbf{K}_b]_{i,j} = k_b(\mathbf{x}_i, \mathbf{x}_j) = b(\mathbf{x}_i) b(\mathbf{x}_j) k_e(\mathbf{x}_i, \mathbf{x}_j)$ and $\sigma_n^2[f]$ is a scale estimator, to be specified.

Given an explicit error bound b such that $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ and $b(\mathbf{0}) = 0$.

How to encode this knowledge into a probabilistic regression model?

Gauss–Richardson Extrapolation (GRE)

Samples $g \sim \mathcal{GP}(0, k)$ from a Gaussian process with covariance kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \left\{ k_0^2 + b(\mathbf{x}) b(\mathbf{x}') k_e(\mathbf{x}, \mathbf{x}') \right\}, \qquad \mathbf{x}, \mathbf{x}' \in (0, \infty)^d,$$

a.s. satisfy $g(\mathbf{x}) - g(\mathbf{0}) = O(b(\mathbf{x}))$ as well.

So we use this GP regression model, trained on $[f(X_n)]_i = f(\mathbf{x}_i)$, to predict $f(\mathbf{0})$.

$\implies$ Only need to work with *linear* information!

Since k_0^2 is proportional to the prior variance for $f(\mathbf{0})$, we seek to let $k_0^2 \to \infty$, representing the flat prior limit / universal kriging:

$$f(\mathbf{0}) | f(X_n) \sim \mathcal{N}(m_n[f], v_n[f]), \qquad m_n[f] = \frac{\mathbf{1}^\top \mathbf{K}_b^{-1} f(X_n)}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}}, \qquad v_n[f] = \frac{\sigma_n^2[f]}{\mathbf{1}^\top \mathbf{K}_b^{-1} \mathbf{1}},$$

where $[\mathbf{K}_b]_{i,j} = k_b(\mathbf{x}_i, \mathbf{x}_j) = b(\mathbf{x}_i) b(\mathbf{x}_j) k_e(\mathbf{x}_i, \mathbf{x}_j)$ and $\sigma_n^2[f]$ is a scale estimator, to be specified.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

- Let $A \in \mathbb{R}^{m \times m}$.
- An *eigenvalue* $\lambda \in \mathbb{R}$ satisfies $Av = \lambda v$ for some $v \in \mathbb{R}^m$.
- All such matrices have n (possibly complex or repeated) eigenvalues, say $\{\lambda_1, \ldots, \lambda_m\}$.
- Eigenvalues are important in many applications, e.g.
  - stability analysis of dynamical systems,
  - web search engines,
  - principal component analysis, ...
- The basic idea of the **QR Algorithm** is to set $A_0 = A$ and iterate

$$A_{n-1} = Q_{n-1} R_{n-1}, \qquad A_n = R_{n-1} Q_{n-1}.$$

  where Q_{n-1} is orthogonal and R_{n-1} is upper-triangular. The diagonal of A_k converges to the set of eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$.
- To see why this works some calculation is needed.
- Also lots of details that are practically important.

Nonlinear information $\implies$ difficult to envisage as a probabilistic numerical method.

Illustration: Eigenvalue Problems

Example: Sparse matrices that arise as the discrete Laplace operator in the solution of the Poisson equation by a finite difference method with a five-point stencil

$$A = \begin{pmatrix} B & -I & & \\ -I & B & -I & \\ & \ddots & \ddots & -I \\ & & -I & B \end{pmatrix}, \qquad B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{pmatrix},$$

where B is an $l \times l$ matrix and A is an $ml \times ml$ matrix, and we aim to recover the largest few eigenvalues of the matrices considered.

For GRE we took:

- $x_n = 1/n$, where n is the number of iterations performed.
- entries of A_n are modelled as *a priori* independent; $f(x_n) = [A_n]_{i,i}$ (but this might be naïve).
- the order s of convergence depends on $\{\lambda_1, \ldots, \lambda_m\}$, so is presumed *unknown* and estimated.

Illustration: Eigenvalue Problems

Example: Sparse matrices that arise as the discrete Laplace operator in the solution of the Poisson equation by a finite difference method with a five-point stencil

$$A = \begin{pmatrix} B & -I & & \\ -I & B & -I & \\ & \ddots & \ddots & -I \\ & & -I & B \end{pmatrix}, \qquad B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{pmatrix},$$

where B is an $l \times l$ matrix and A is an $ml \times ml$ matrix, and we aim to recover the largest few eigenvalues of the matrices considered.

For GRE we took:

- $x_n = 1/n$, where n is the number of iterations performed.
- entries of A_n are modelled as *a priori* independent; $f(x_n) = [A_n]_{i,i}$ (but this might be naïve).
- the order s of convergence depends on $\{\lambda_1, \ldots, \lambda_m\}$, so is presumed *unknown* and estimated.

Illustration: Eigenvalue Problems

Example: Sparse matrices that arise as the discrete Laplace operator in the solution of the Poisson equation by a finite difference method with a five-point stencil

$$A = \begin{pmatrix} B & -I & & \\ -I & B & -I & \\ & \ddots & \ddots & -I \\ & & -I & B \end{pmatrix}, \qquad B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{pmatrix},$$

where B is an $l \times l$ matrix and A is an $ml \times ml$ matrix, and we aim to recover the largest few eigenvalues of the matrices considered.

For GRE we took:

- $x_n = 1/n$, where n is the number of iterations performed.
- entries of A_n are modelled as *a priori* independent; $f(x_n) = [A_n]_{i,i}$ (but this might be naïve).
- the order s of convergence depends on $\{\lambda_1, \ldots, \lambda_m\}$, so is presumed *unknown* and estimated.

Illustration: Eigenvalue Problems

Example: Sparse matrices that arise as the discrete Laplace operator in the solution of the Poisson equation by a finite difference method with a five-point stencil

$$A = \begin{pmatrix} B & -I & & \\ -I & B & -I & \\ & \ddots & \ddots & -I \\ & & -I & B \end{pmatrix}, \qquad B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{pmatrix},$$

where B is an $l \times l$ matrix and A is an $ml \times ml$ matrix, and we aim to recover the largest few eigenvalues of the matrices considered.

For GRE we took:

- $x_n = 1/n$, where n is the number of iterations performed.
- entries of A_n are modelled as *a priori* independent; $f(x_n) = [A_n]_{i,i}$ (but this might be naïve).
- the order s of convergence depends on $\{\lambda_1, \ldots, \lambda_m\}$, so is presumed *unknown* and estimated.

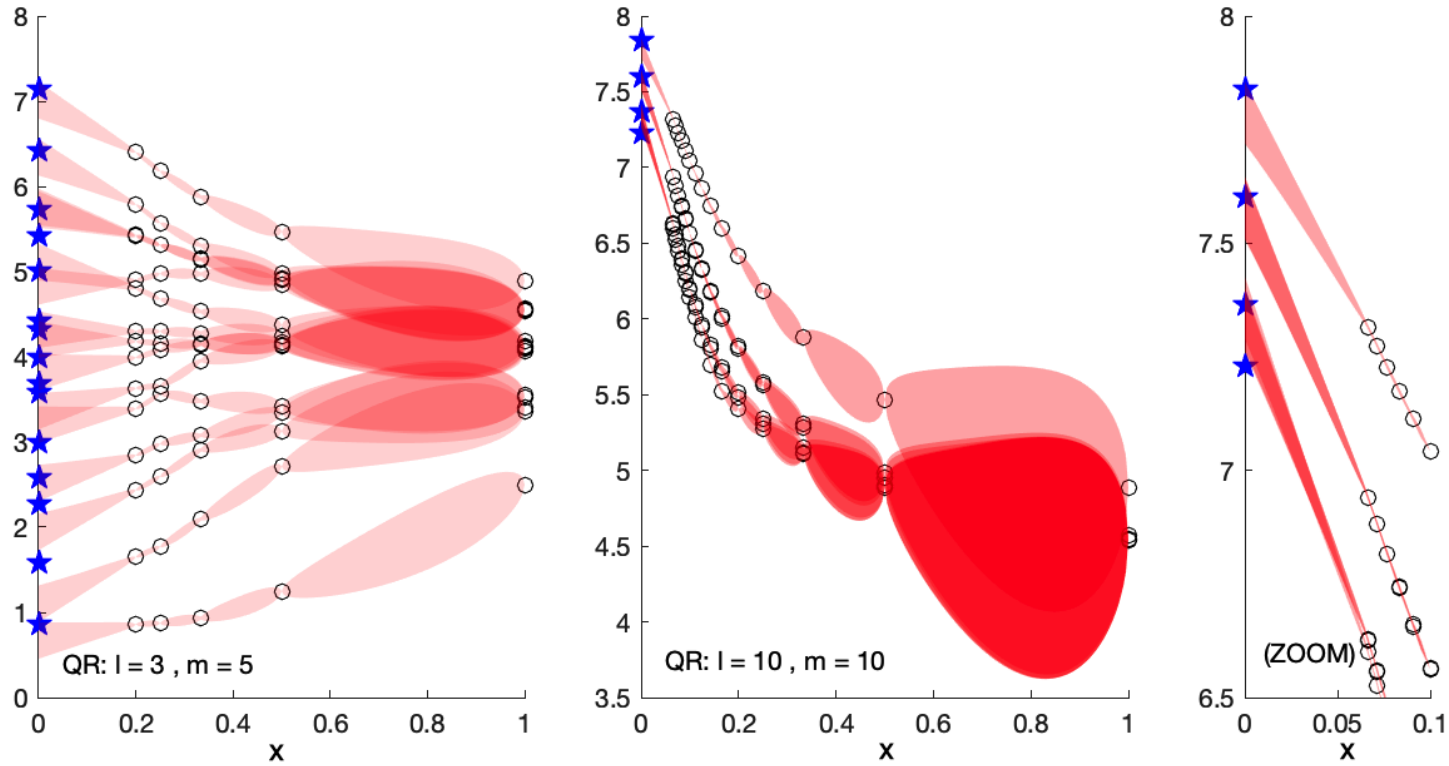


Figure: QR algorithm. All plots show red shaded $\pm 2\sigma$ credible intervals, numerical data as black circles, and true eigenvalues as blue stars. A total of $n = 5$ (left) and 15 (centre) iterations were used.

(✓) No additional computational cost to GRE, since the sequence $(f(x_i))_{i=1}^{n}$ is generated during a single run of the iterative numerical method.

(✓) Overhead due to fitting the Gaussian process (GP) is negligible in this experiment.

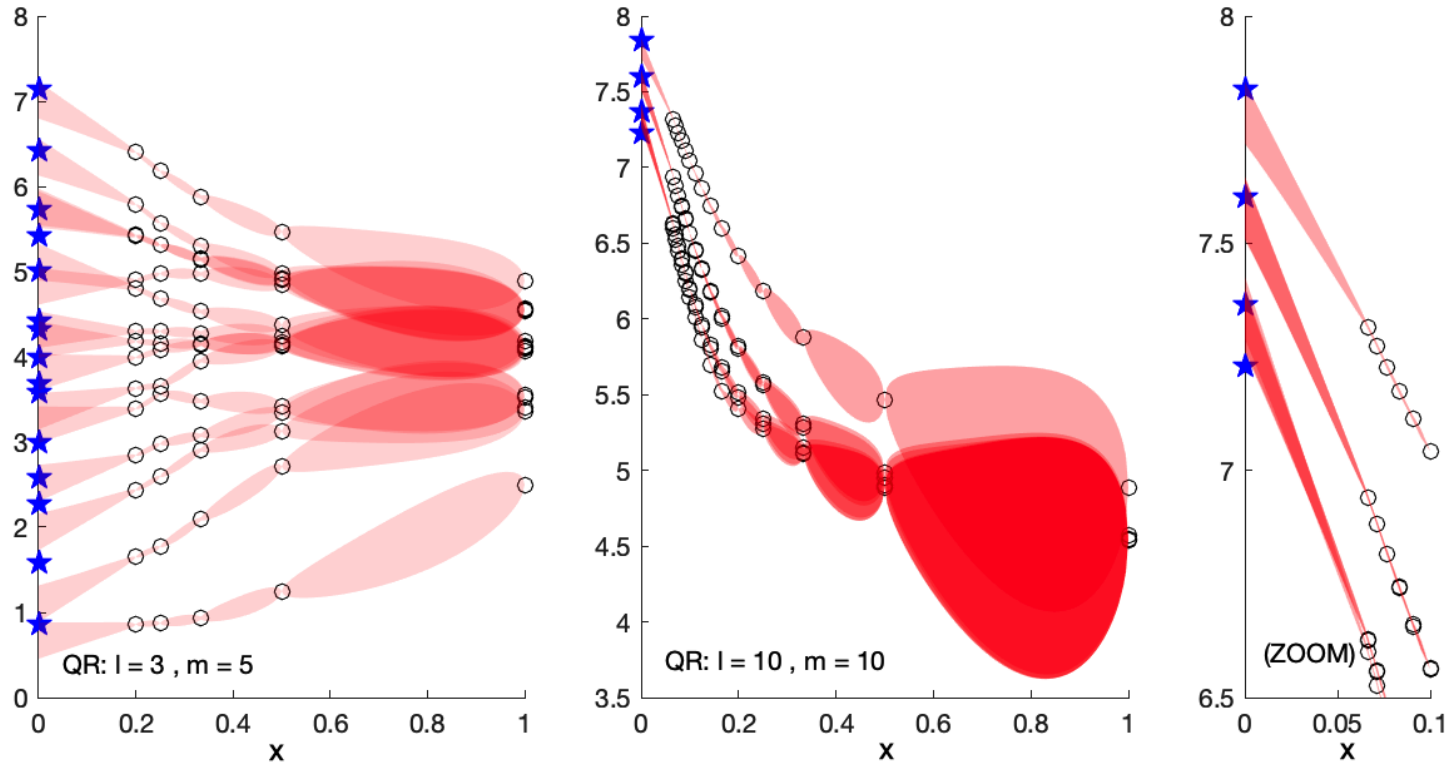


Figure: QR algorithm. All plots show red shaded $\pm 2\sigma$ credible intervals, numerical data as black circles, and true eigenvalues as blue stars. A total of $n = 5$ (left) and 15 (centre) iterations were used.

(✓) No additional computational cost to GRE, since the sequence $(f(x_i))_{i=1}^{n}$ is generated during a single run of the iterative numerical method.

(✓) Overhead due to fitting the GP is negligible in this experiment.

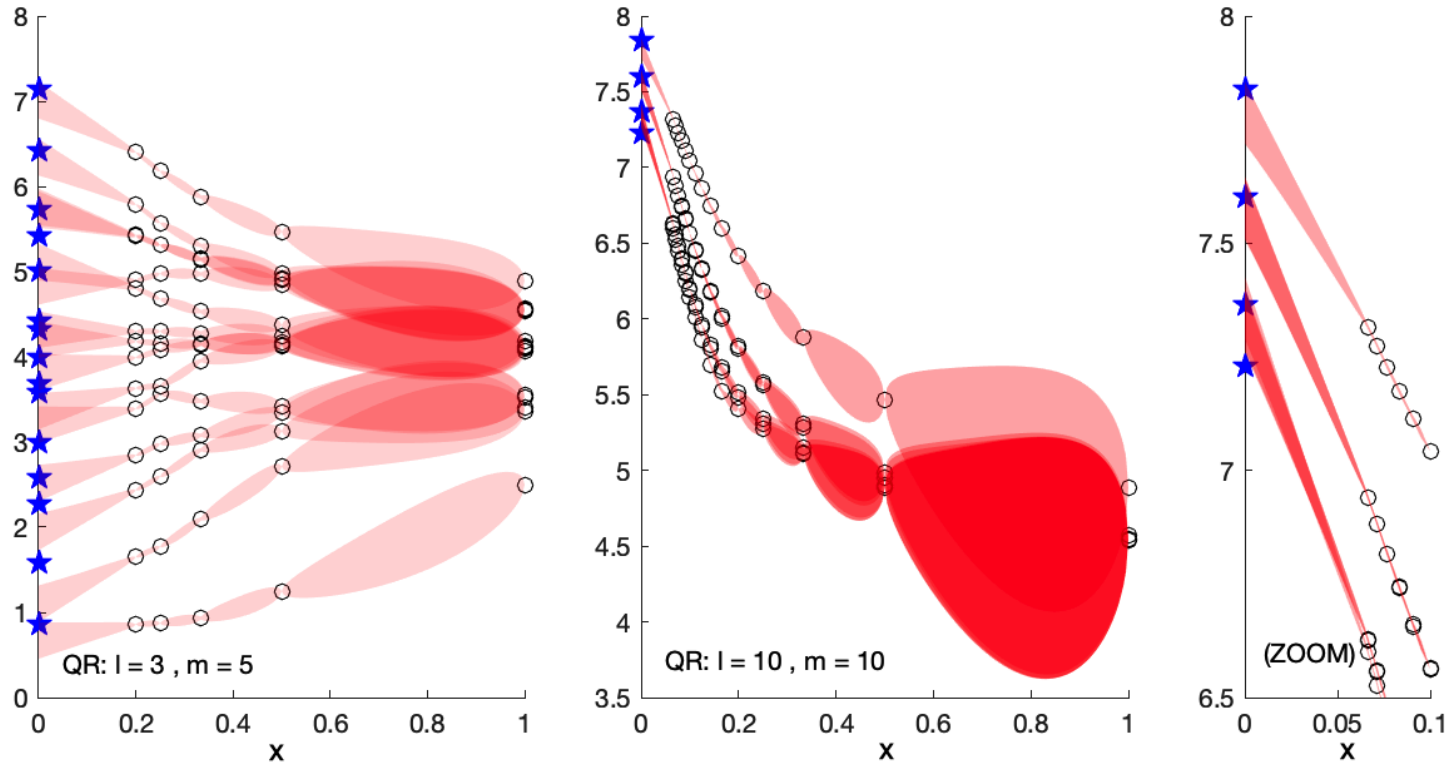


Figure: QR algorithm. All plots show red shaded $\pm 2\sigma$ credible intervals, numerical data as black circles, and true eigenvalues as blue stars. A total of $n = 5$ (left) and 15 (centre) iterations were used.

(✓) No additional computational cost to GRE, since the sequence $(f(x_i))_{i=1}^{n}$ is generated during a single run of the iterative numerical method.

(✓) Overhead due to fitting the GP is negligible in this experiment.

Importance of a Non-Stationary GP: Recall that s is *inferred* in these simulations - the estimated values were, respectively, 1.0186 and 1.0167.

Contrast with a *stationary* GP model (*i.e.* $s = 0$):

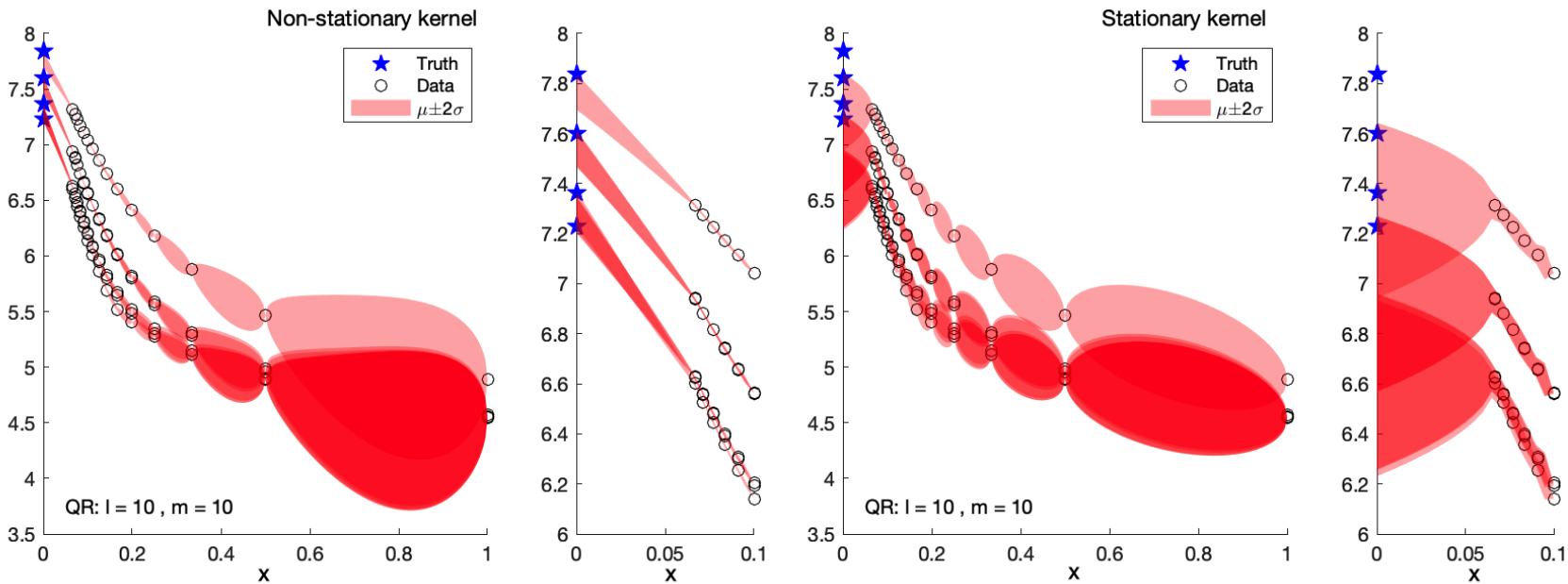


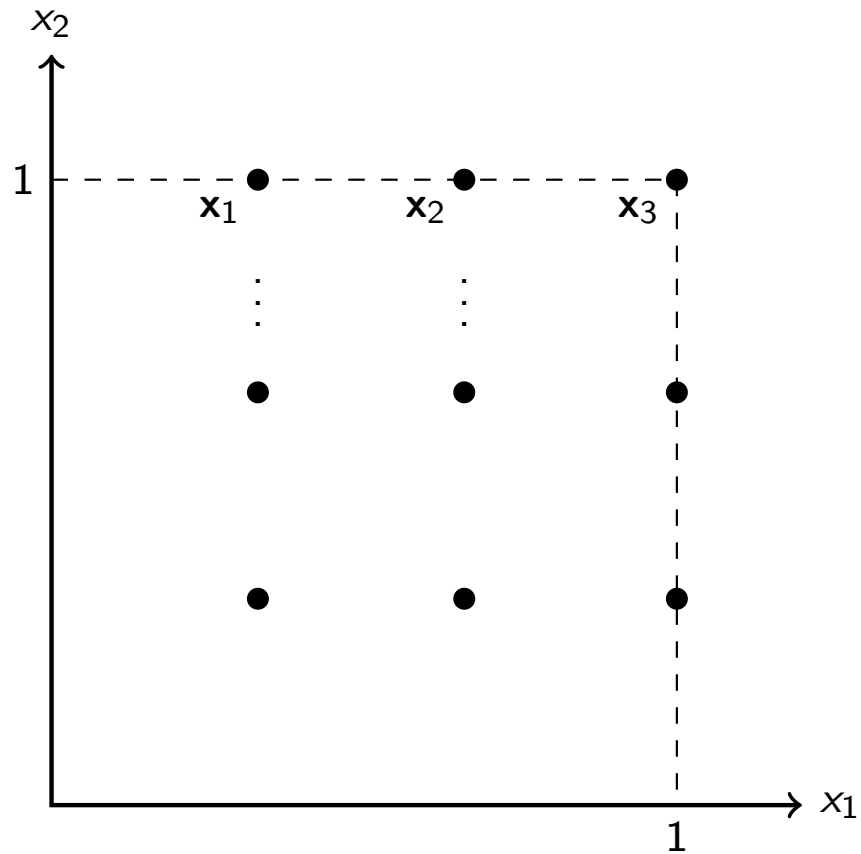
Figure: Comparison of non-stationary (left) and stationary (right) covariance kernels.

Theory of Gauss–Richardson Extrapolation

Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

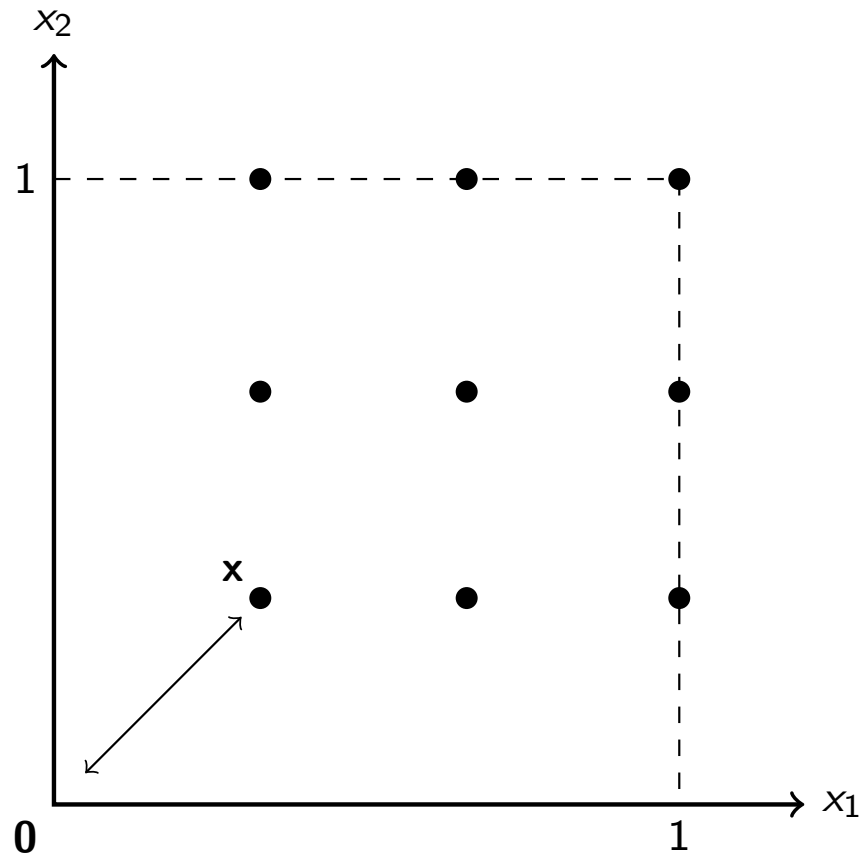
Design points $X_n = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset [0,1]^d$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

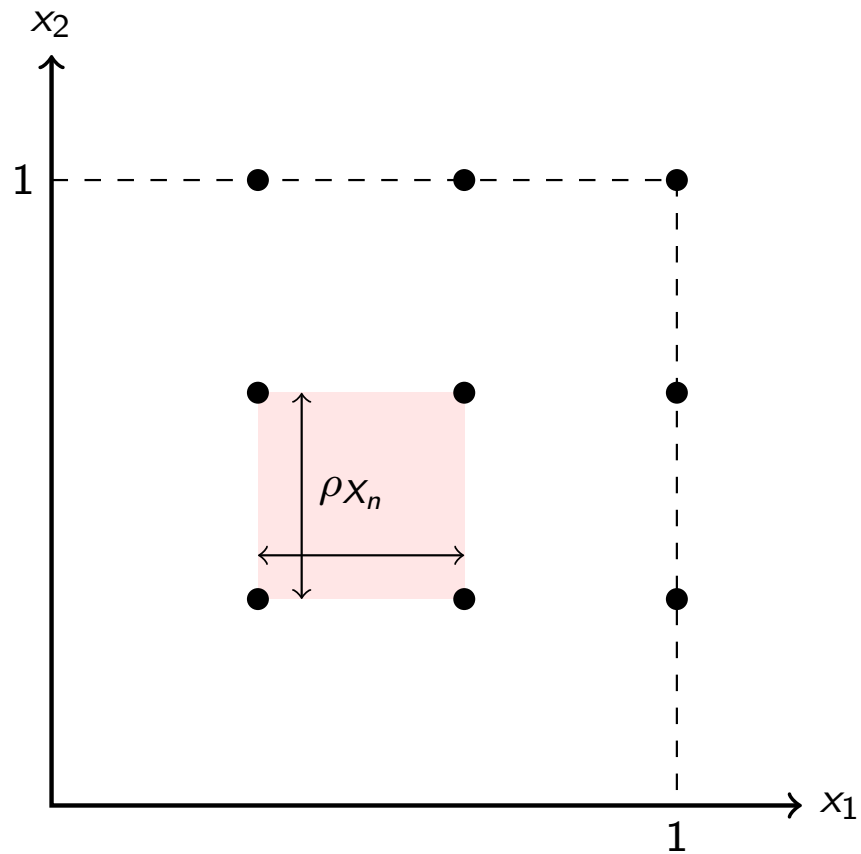
Error of the highest-fidelity experiment is $f(\mathbf{x}) - f(\mathbf{0}) = O(b(\mathbf{x}))$



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

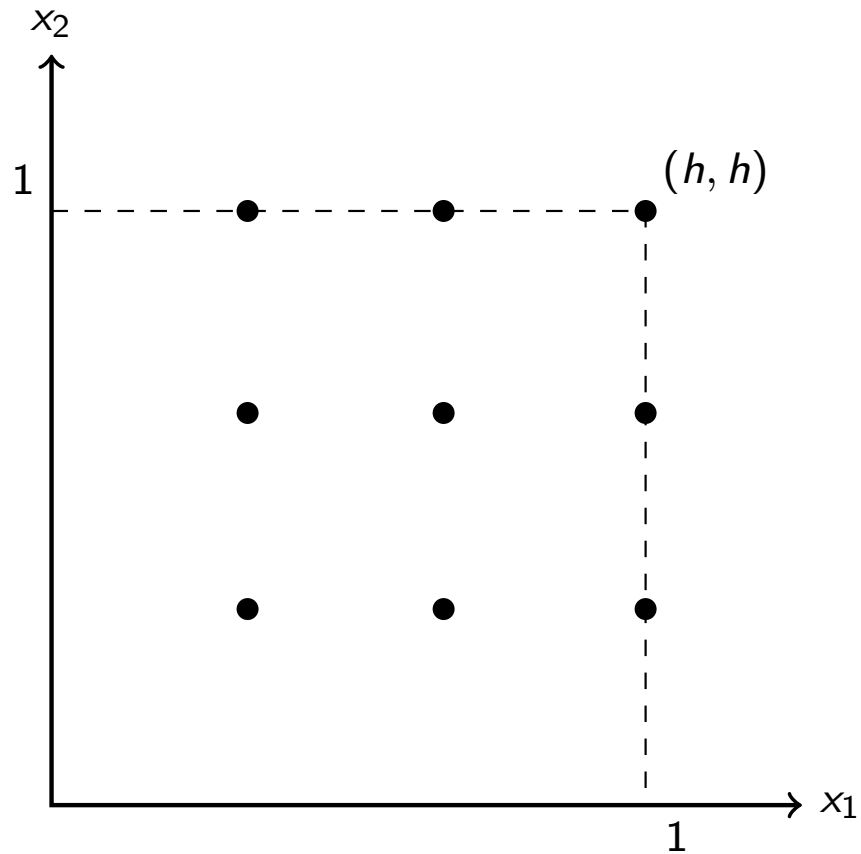
Box fill distance ρ_{X_n} = size of the biggest cube not containing an element of X_n.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

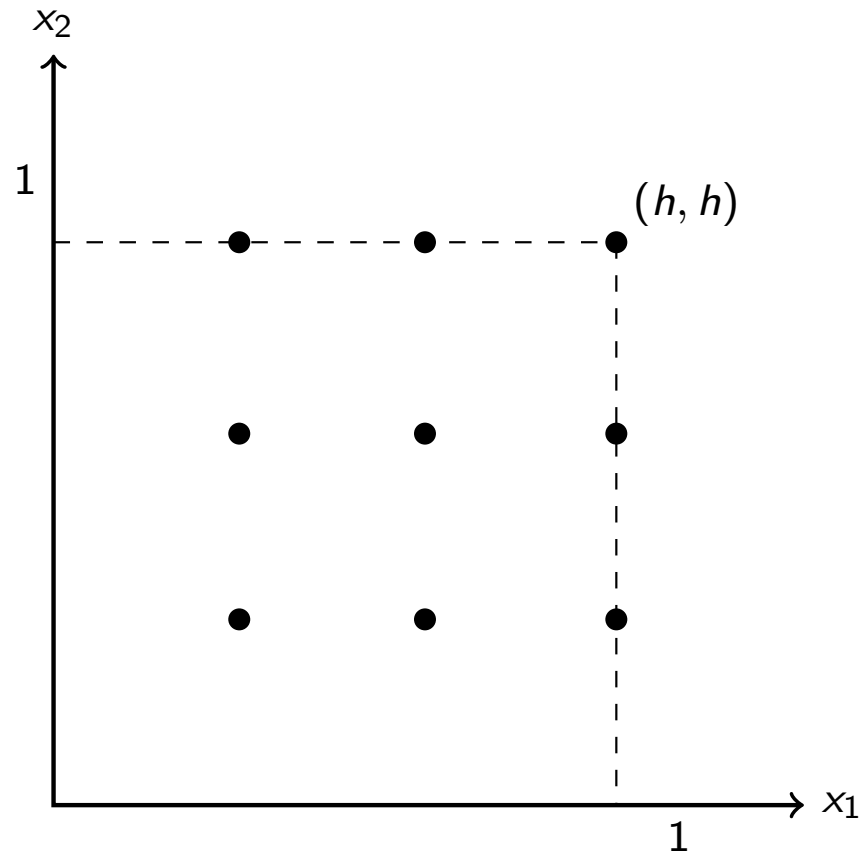
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

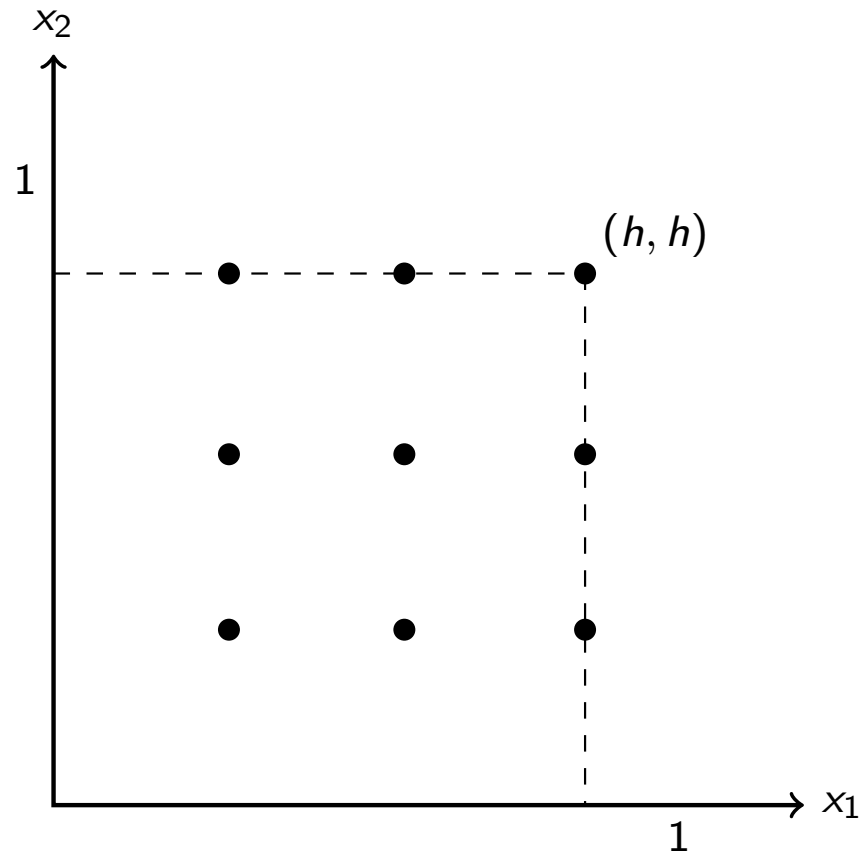
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

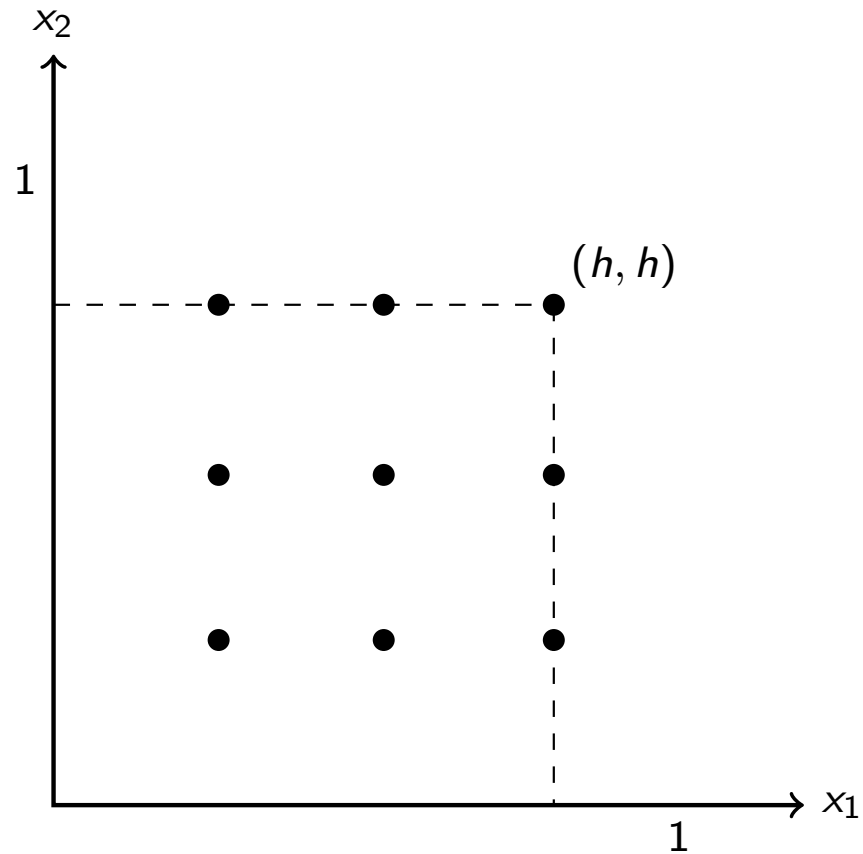
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

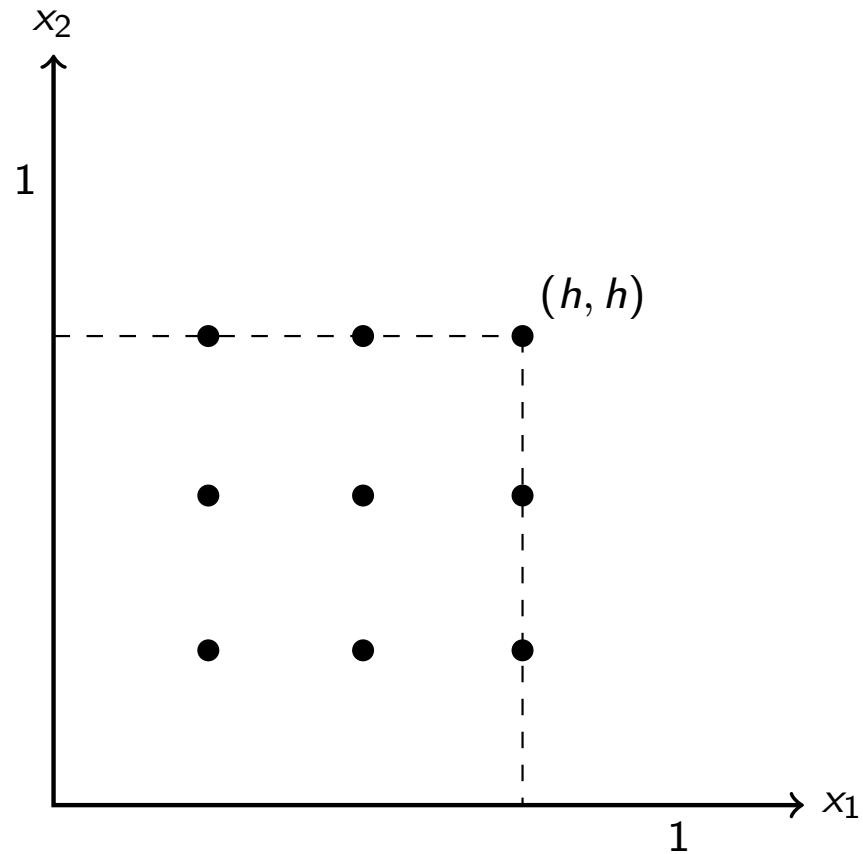
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

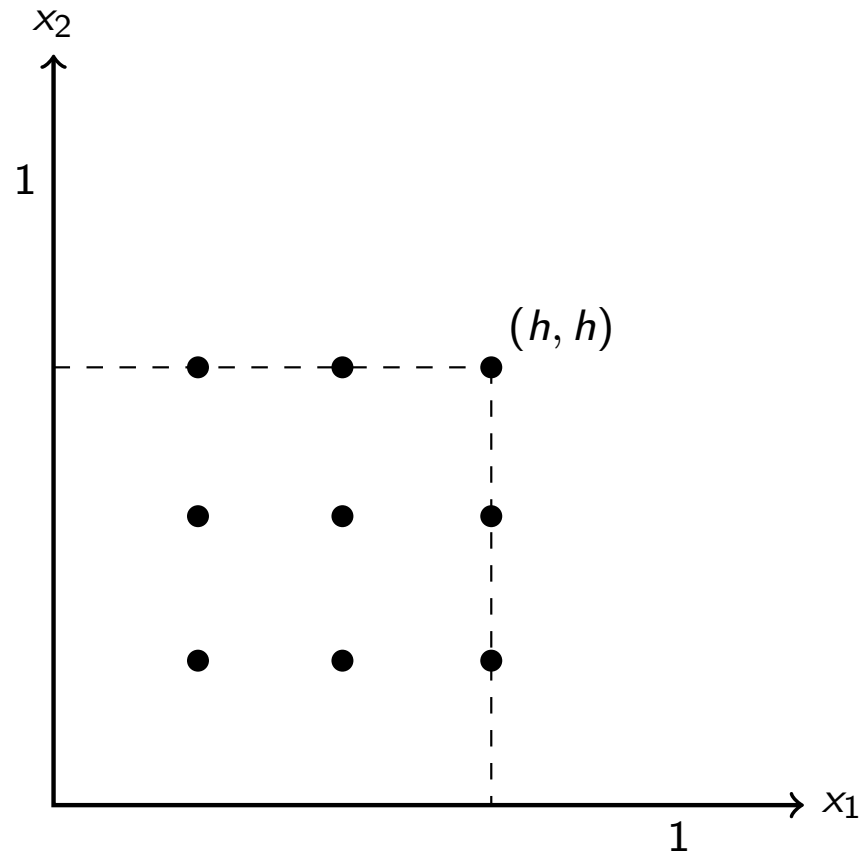
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

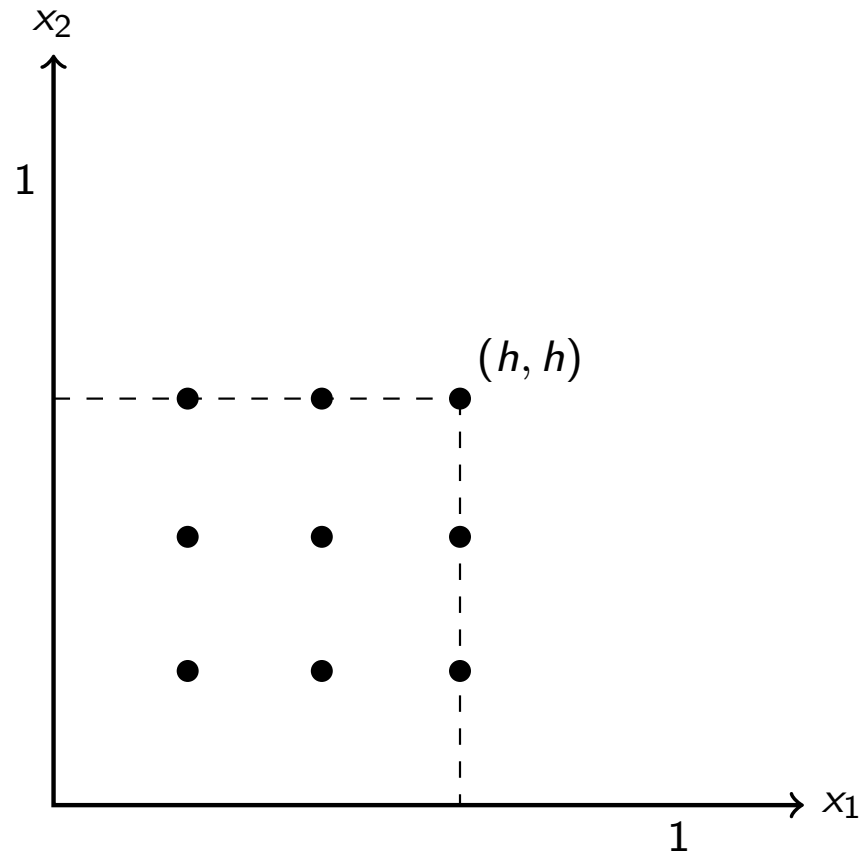
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

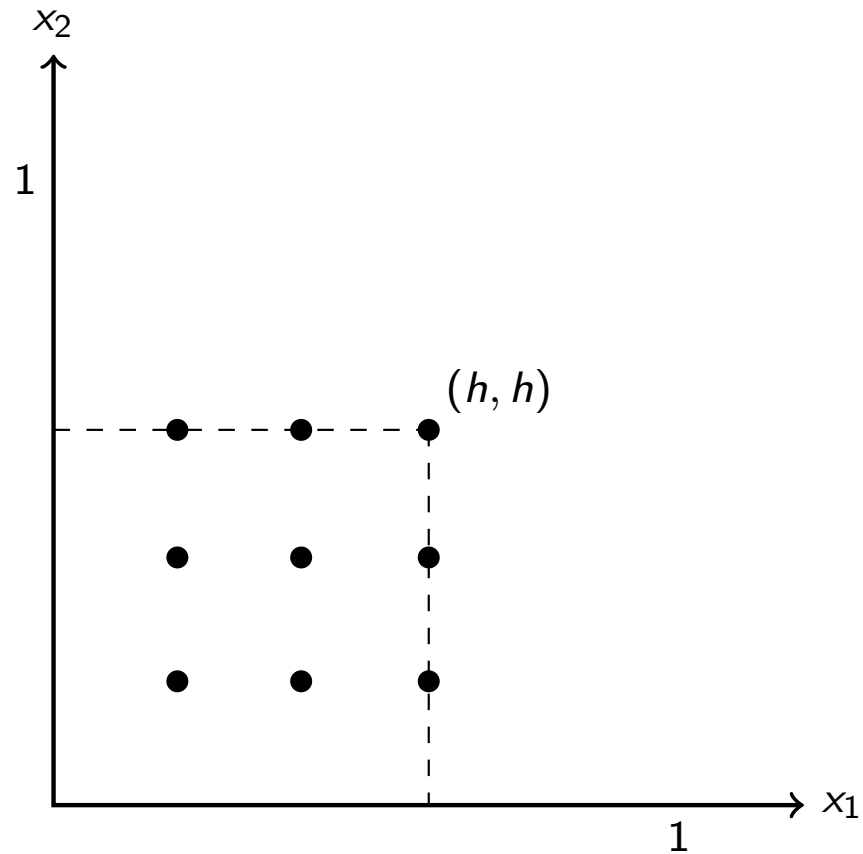
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

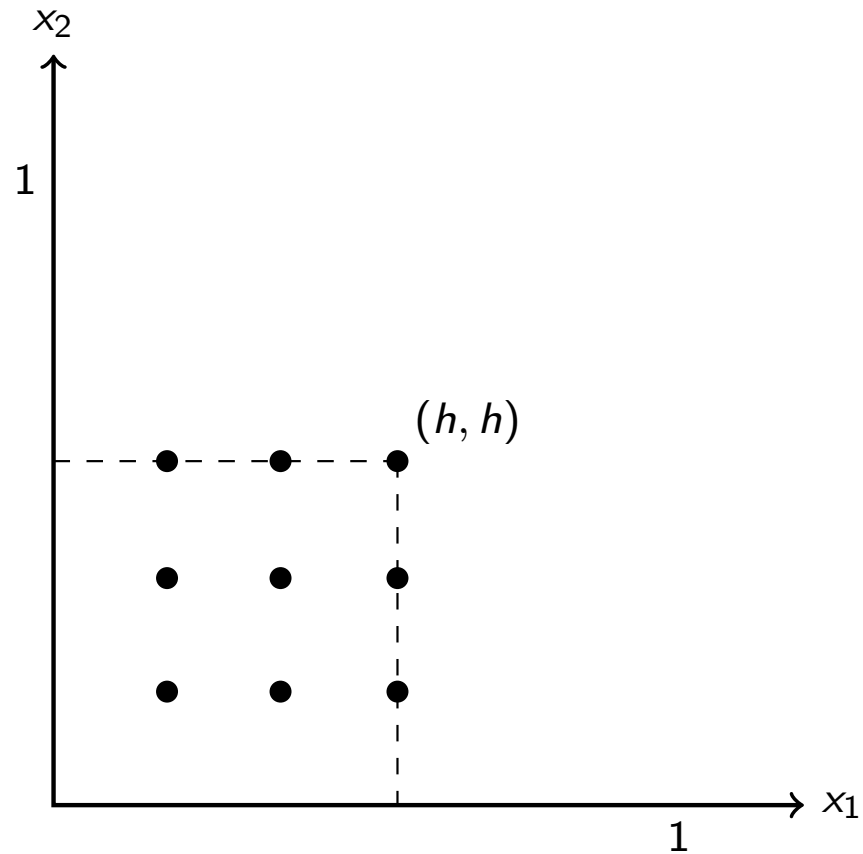
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

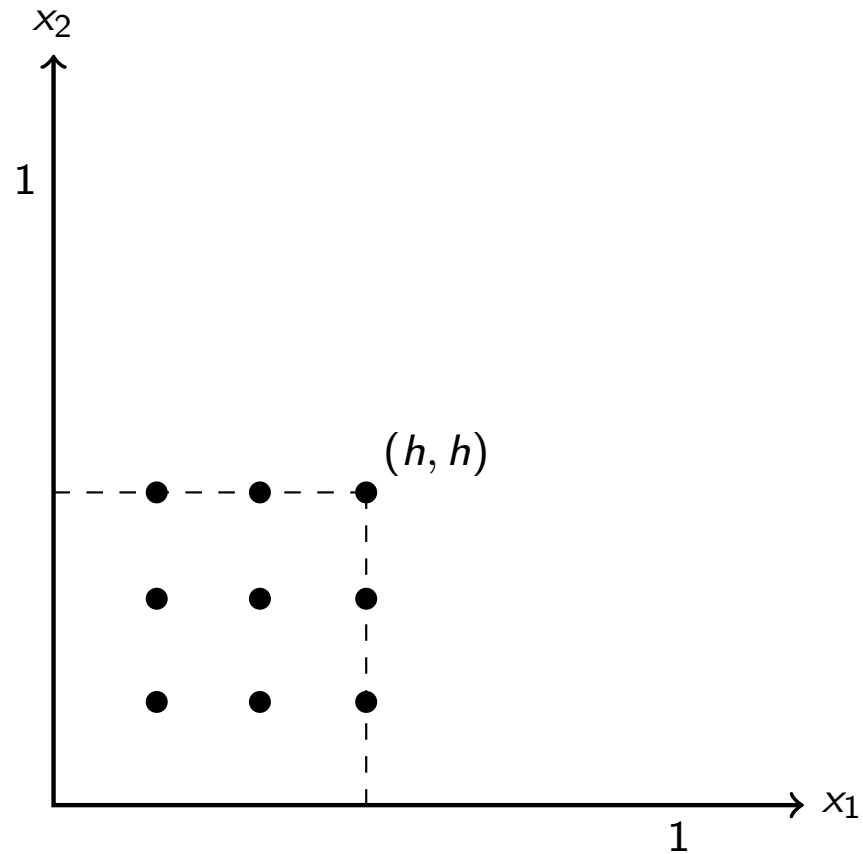
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

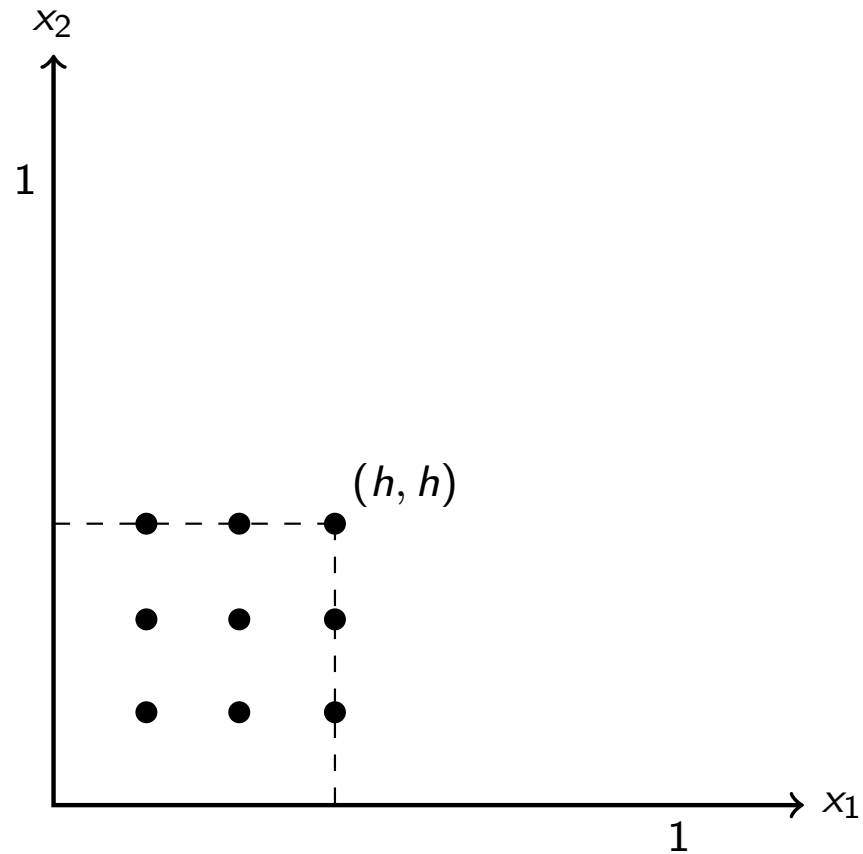
Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Set-up for theoretical analysis of GRE

Recall $f(\mathbf{x}) = f(\mathbf{0}) + O(b(\mathbf{x}))$ for some $\mathbf{x} \in [0,1]^d$.

Scaled design points $X_n^h = \{h\boldsymbol{x} : \boldsymbol{x} \in X_n\}$ where $h \in (0,1]$.



Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{|f(\mathbf{0}) - m_n^h[f]|}_{\text{extrapolation error}} \leq C_s \, \rho_{X_n}^s |f|_{\mathcal{H}_k(\mathcal{X})} \underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}}$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \tfrac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{|f(\mathbf{0}) - m_n^h[f]|}_{\text{extrapolation error}} \leq C_s \, \rho_{X_n}^s |f|_{\mathcal{H}_k(\mathcal{X})} \underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}}$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \tfrac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{|f(\mathbf{0}) - m_n^h[f]|}_{\text{extrapolation error}} \leq C_s \, \rho_{X_n}^s |f|_{\mathcal{H}_k(\mathcal{X})} \underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}}$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \tfrac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{|f(\mathbf{0}) - m_n^h[f]|}_{\text{extrapolation error}} \leq C_s \, \rho_{X_n}^s |f|_{\mathcal{H}_k(\mathcal{X})} \underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}}$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \frac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{|f(\mathbf{0}) - m_n^h[f]|}_{\text{extrapolation error}} \leq C_s \, \rho_{X_n}^s |f|_{\mathcal{H}_k(\mathcal{X})} \underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}}$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \tfrac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{f(\mathbf{0}) - m_n^h[f]}_{\text{extrapolation error}} = O\Big(\underbrace{h^s}_{\text{acceleration}} \quad \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}} \Big)$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \tfrac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{f(\mathbf{0}) - m_n^h[f]}_{\text{extrapolation error}} = O\Big(\underbrace{h^s}_{\text{acceleration}} \quad \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}} \Big)$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Proof via local polynomial reproduction, *a la* Wendland [2004].
- Applies to s-smooth f, meaning that

$$|f|_{\mathcal{H}_k(\mathcal{X})} = \left\| \mathbf{x} \mapsto e(\mathbf{x}) := \tfrac{f(\mathbf{x}) - f(\mathbf{0})}{b(\mathbf{x})} \right\|_{\mathcal{H}_{k_e}(\mathcal{X})} < \infty.$$

- Extrapolation with convergence rate error $O(h^{s+r})$.
- Analogous result with exponential rates when f is smooth enough.

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{f(\mathbf{0}) - m_n^h[f]}_{\text{extrapolation error}} = O\Big(\underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}} \Big)$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Need
$$\rho_{X_n} \leq \frac{1}{\gamma_d(r+2s)},$$
where $\gamma_d := 2d(1+\gamma_{d-1})$ with base case $\gamma_1 := 2$.
- Question: We lose the "one order per datum" of Richardson - is something lost in the proof?

Theorem (Higher-order convergence with GRE)

Assume that $b \in \mathrm{poly}(r)$ *and* $k_e \in C^{2s}$.

Let $m_n^h[f]$ *be the GRE estimator based on* $f(X_n^h)$.

Then there is an explicit constant C_s *such that*

$$\underbrace{f(\mathbf{0}) - m_n^h[f]}_{\text{extrapolation error}} = O\Big(\underbrace{h^s}_{\text{acceleration}} \underbrace{\|b\|_{L^\infty([0,h]^d)}}_{\text{original bound}} \Big)$$

whenever the box fill distance ρ_{X_n} *is "small enough".*

Remarks:

- Need
$$\rho_{X_n} \leq \frac{1}{\gamma_d(r+2s)},$$
where $\gamma_d := 2d(1+\gamma_{d-1})$ with base case $\gamma_1 := 2$.
- Question: We lose the "one order per datum" of Richardson - is something lost in the proof?

Illustration: Finite difference method

For $g : \mathbb{R} \to \mathbb{R}$ be $s+1$ times continuously differentiable in an open neighbourhood of $t \in \mathbb{R}$, and consider the *central difference* method

$$\mathrm{f}(x) = \frac{g(t+x) - g(t-x)}{2x}$$

for approximation of $g'(t)$.

Central differences are second-order accurate, so we take $b(x) = x^2$.

$X_n^h = \{0.2h, 0.4h, 0.6h, 0.8h, h\}$.

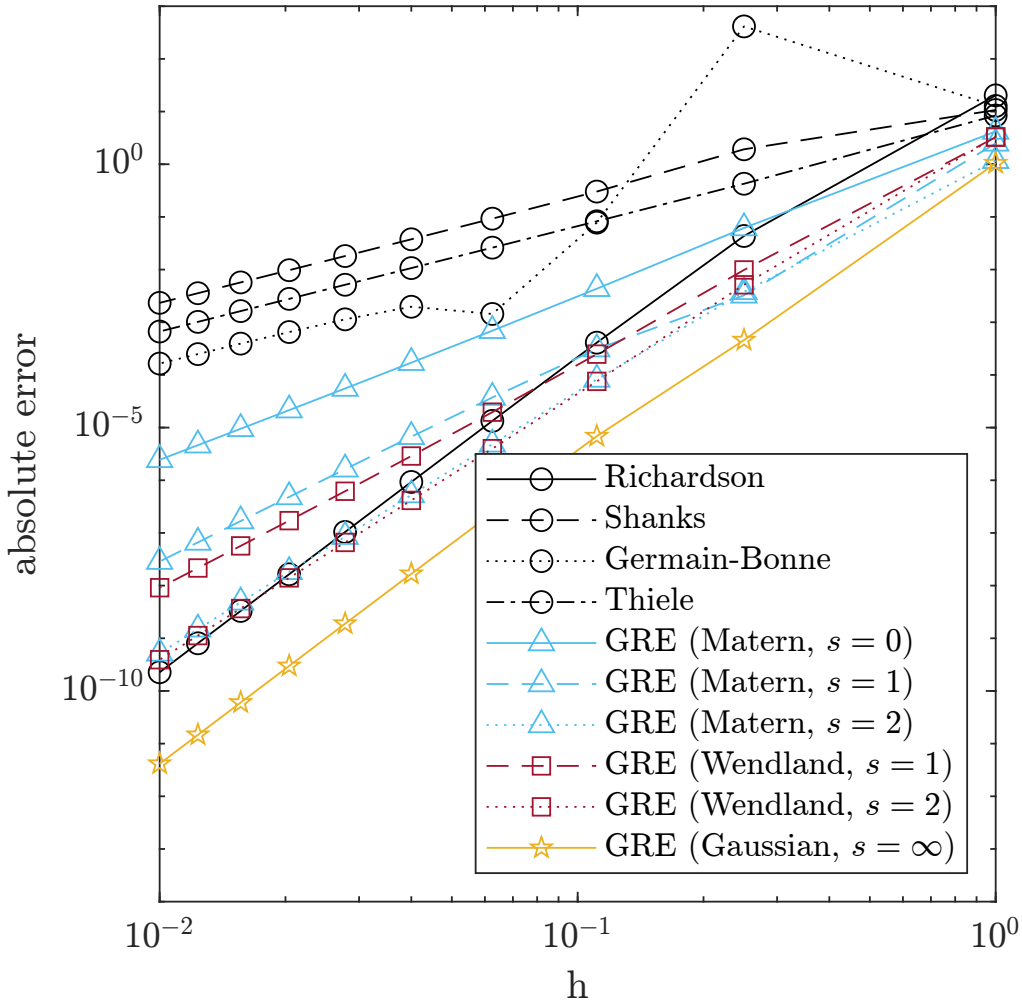


Figure: Finite difference approximation; $g \in C^3$ $(s = 2)$.

Uncertainty quantification

For the maximum quasi likelihood estimator

$$\sigma_n^2[f] = \frac{1}{n}\Big[f(X_n)^\top \mathbf{K}_b^{-1} f(X_n) - \frac{(\mathbf{1}^\top \mathbf{K}_b^{-1} f(X_n))^2}{\mathbf{1}^\top \mathbf{K}_b^{-1}\mathbf{1}}\Big]$$

we can show that

$$\limsup_{h\to 0} \frac{|f(\mathbf{0}) - m_n^h[f]|}{\sqrt{v_n^h[f]}} < \infty$$

which is nice, but we seem to be a bit asymptotically over-confident when using the "right" kernel ($s = 2$).

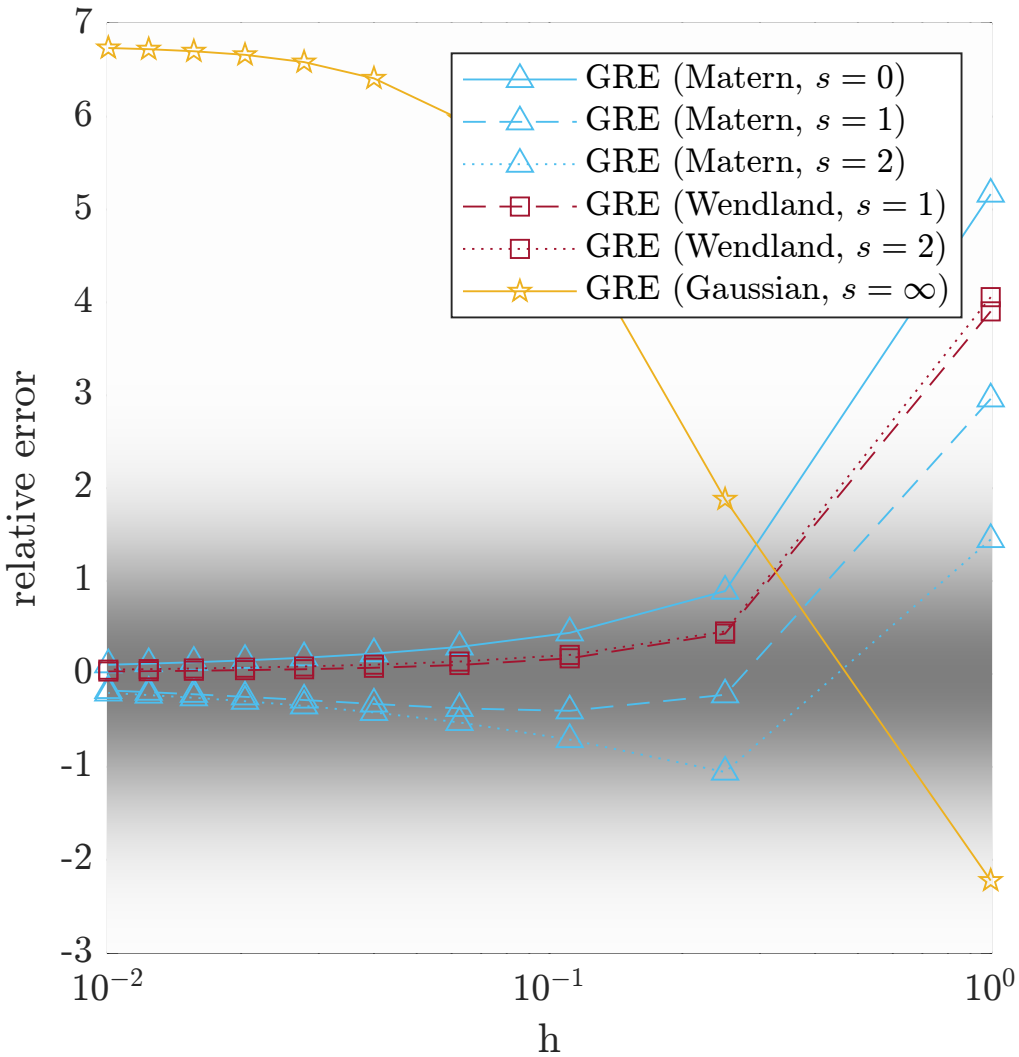


Figure: Finite difference approximation; $g \in C^3$ ($s = 2$)

Optimal experimental design

The variance returned from GRE is

$$v_n[\mathrm{f}] = \frac{\sigma_n^2[\mathrm{f}]}{\mathbf{1}^\top \mathbf{K}_b^{-1}\mathbf{1}}$$

An *a priori* optimal experimental design is

$$\arg\max_X \quad \mathbf{1}^\top \mathbf{K}_b^{-1}\mathbf{1} \text{ s.t. } \sum_{\mathbf{x}\in X} c(\mathbf{x}) \leq C,$$

where $\mathbf{K}_b = [k_b(\boldsymbol{x}, \boldsymbol{x}')]_{\boldsymbol{x},\boldsymbol{x}'\in X}$.

Let's vary the computational budget C and look at optimal designs:

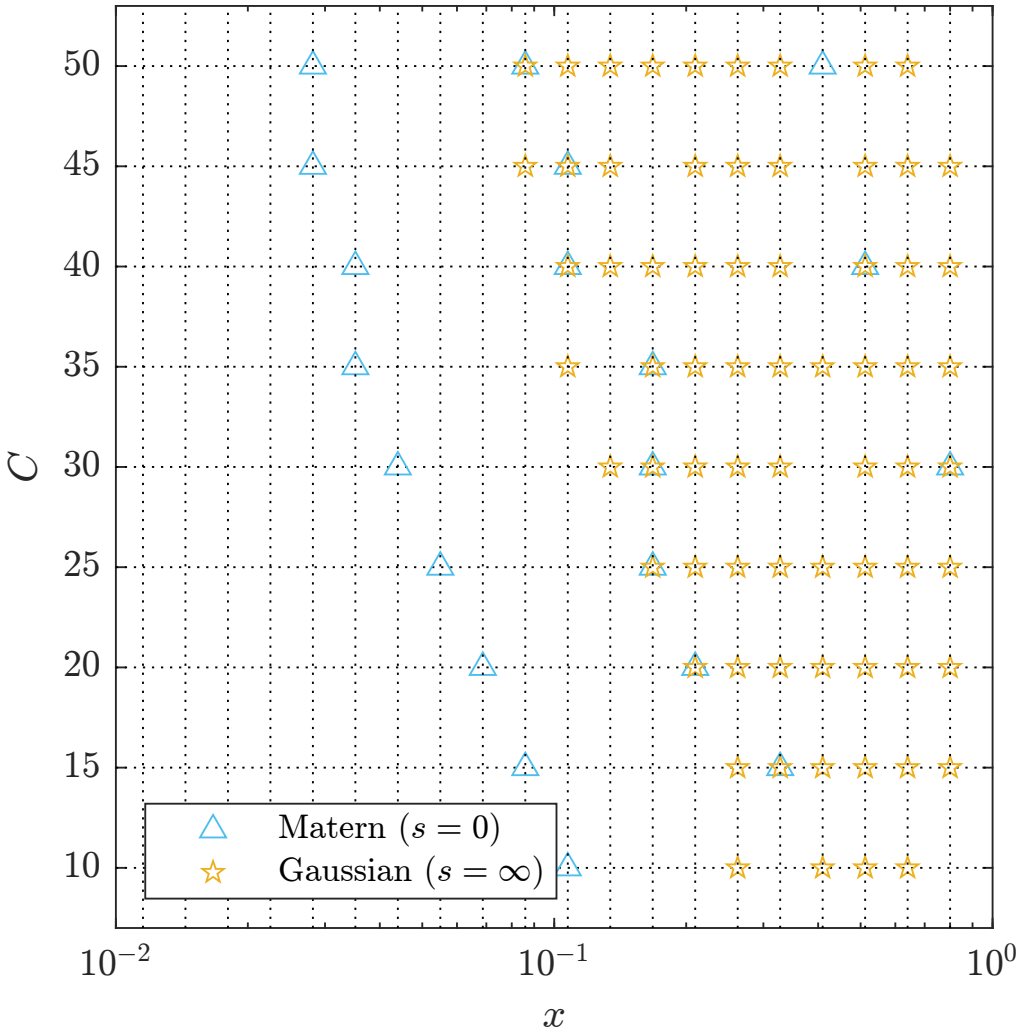


Figure: Optimal experimental design; $b(x) = x$, $c(x) = x^{-1}$

Case Study: Cardiac Modelling

Cardiac Model

Here $f(\mathbf{x})$ is a numerical simulation of a single heart beat with both a spatial (x_1) and a temporal (x_2) discretisation level [Strocchi et al., 2023].

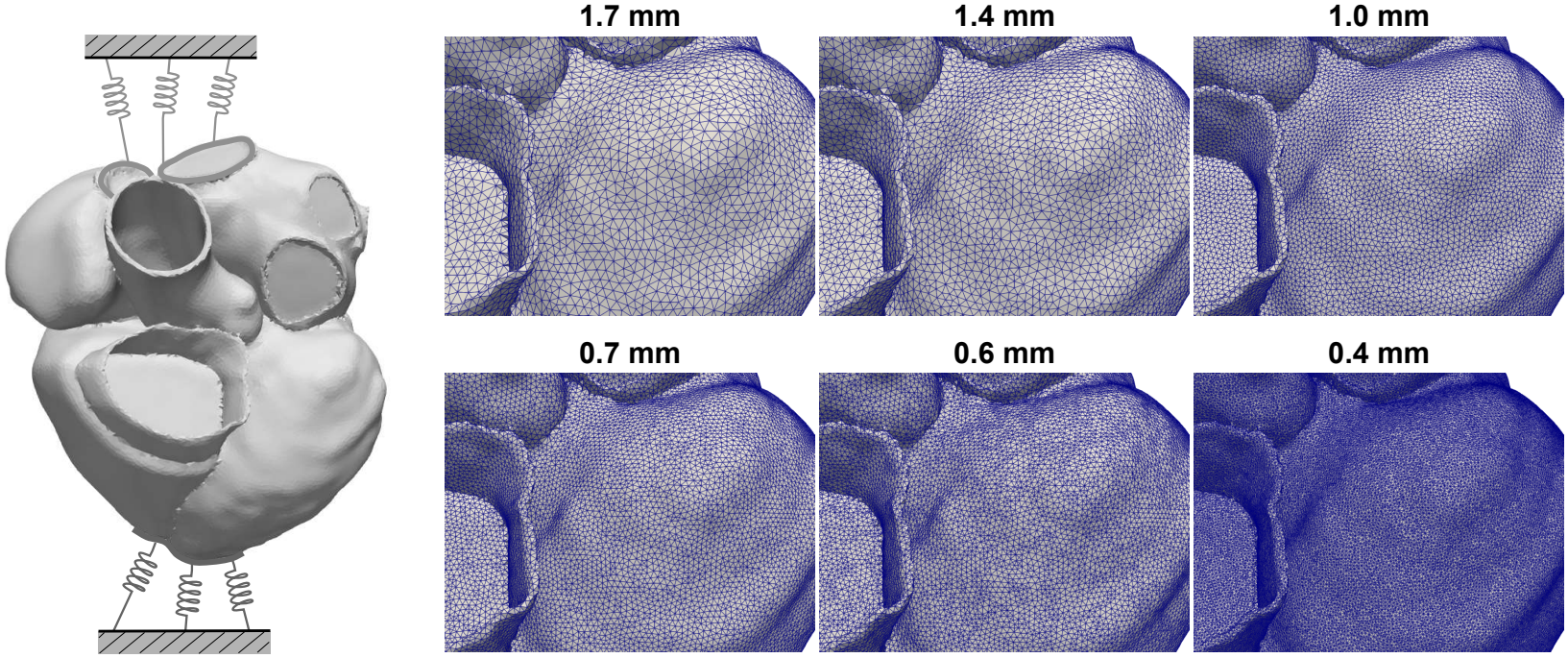


Figure: Cardiac model: A subset of the mesh resolutions used in this case study. The finest resolution required 3×10^7 finite elements to be used.

The computational cost $c(\mathbf{x})$ is measured in real computational time (seconds) and comprises

- setup time
- assembly time (the time taken to assemble linear systems of equations)
- solver time (the time taken to solve linear systems of equations)

with assembly time the main contributor to total computational cost.

To achieve a clinically-acceptable level of accuracy, it is typical for a simulation to be performed with $\mathbf{x}_{\text{default}} \approx (0.4 \text{ mm}, 2 \text{ ms})$, at a cost $c(\mathbf{x}_{\text{default}}) \approx 1.5 \times 10^4$ seconds (around ≈ 4 hours on 512 cores of ARCHER) for a single heart beat.

Here is our workflow:

The computational cost $c(\mathbf{x})$ is measured in real computational time (seconds) and comprises

- setup time
- assembly time (the time taken to assemble linear systems of equations)
- solver time (the time taken to solve linear systems of equations)

with assembly time the main contributor to total computational cost.

To achieve a clinically-acceptable level of accuracy, it is typical for a simulation to be performed with $\mathbf{x}_{\text{default}} \approx (0.4 \text{ mm}, 2 \text{ ms})$, at a cost $c(\mathbf{x}_{\text{default}}) \approx 1.5 \times 10^4$ seconds (around ≈ 4 hours on 512 cores of ARCHER) for a single heart beat.

Here is our workflow:

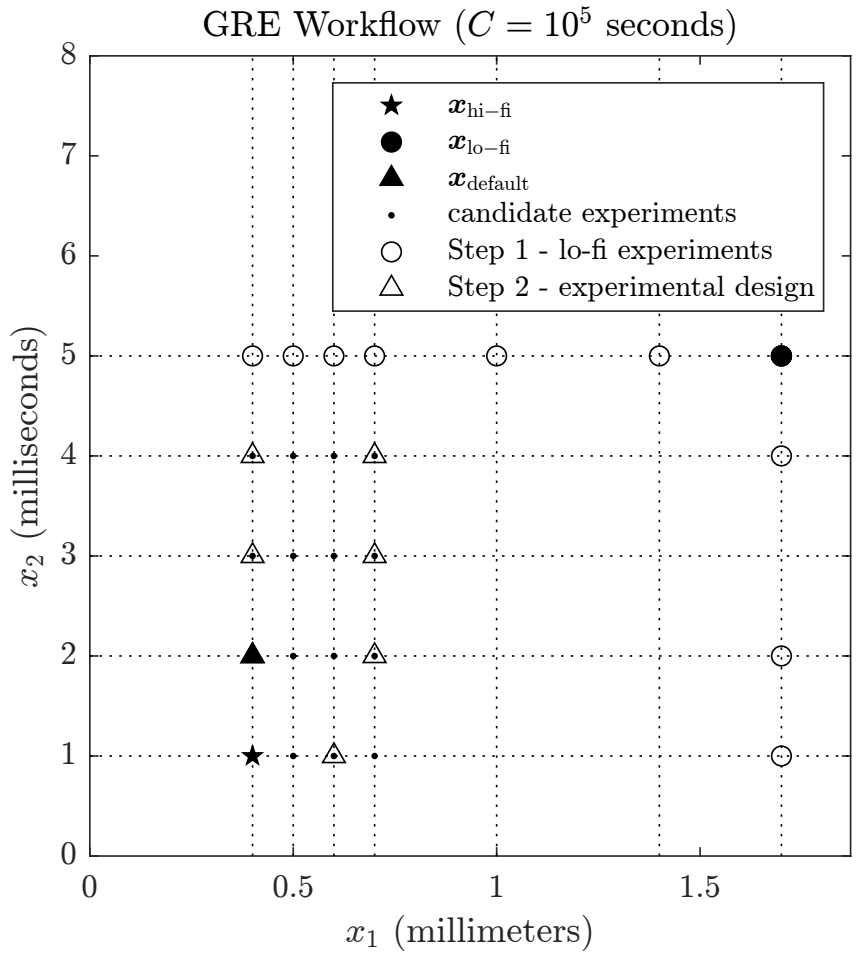
The computational cost $c(\mathbf{x})$ is measured in real computational time (seconds) and comprises

- setup time
- assembly time (the time taken to assemble linear systems of equations)
- solver time (the time taken to solve linear systems of equations)

with assembly time the main contributor to total computational cost.

To achieve a clinically-acceptable level of accuracy, it is typical for a simulation to be performed with $\mathbf{x}_{\text{default}} \approx (0.4 \text{ mm}, 2 \text{ ms})$, at a cost $c(\mathbf{x}_{\text{default}}) \approx 1.5 \times 10^4$ seconds (around ≈ 4 hours on 512 cores of ARCHER) for a single heart beat.

Here is our workflow:



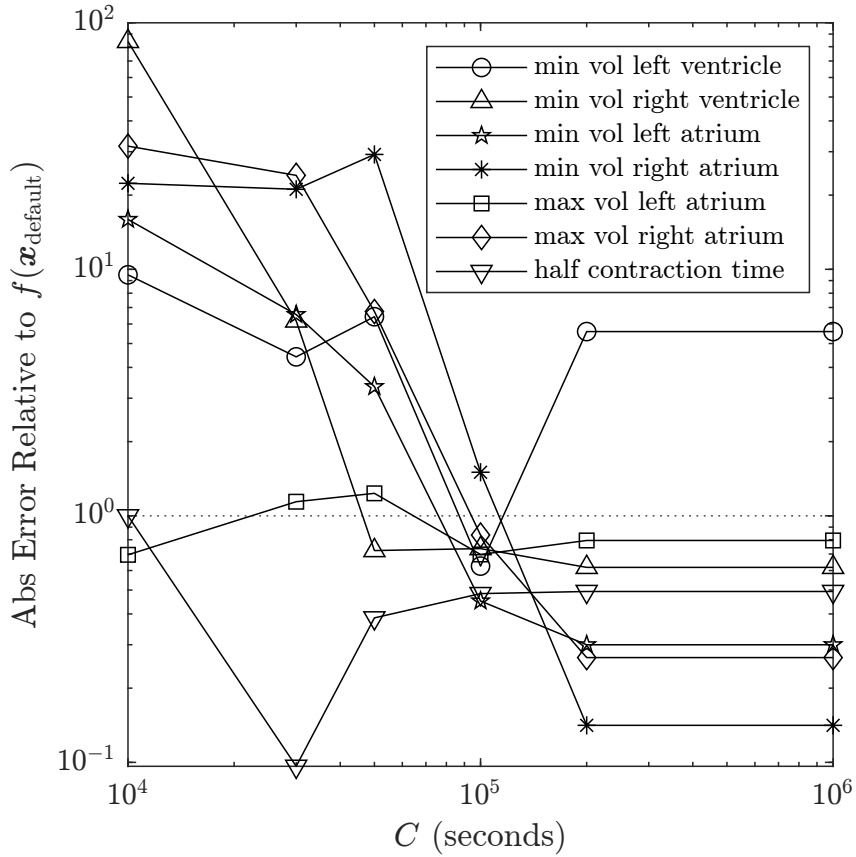
Scalar quantities of interest

For assessment purposes we aim to predict $f(\mathbf{x}_{\text{hi-fi}})$ as a ground truth, but in practice the goal is to predict $f(\mathbf{0})$.

For each of 7 scalar quantities of interest associated with the cardiac model we display the relative error of GRE

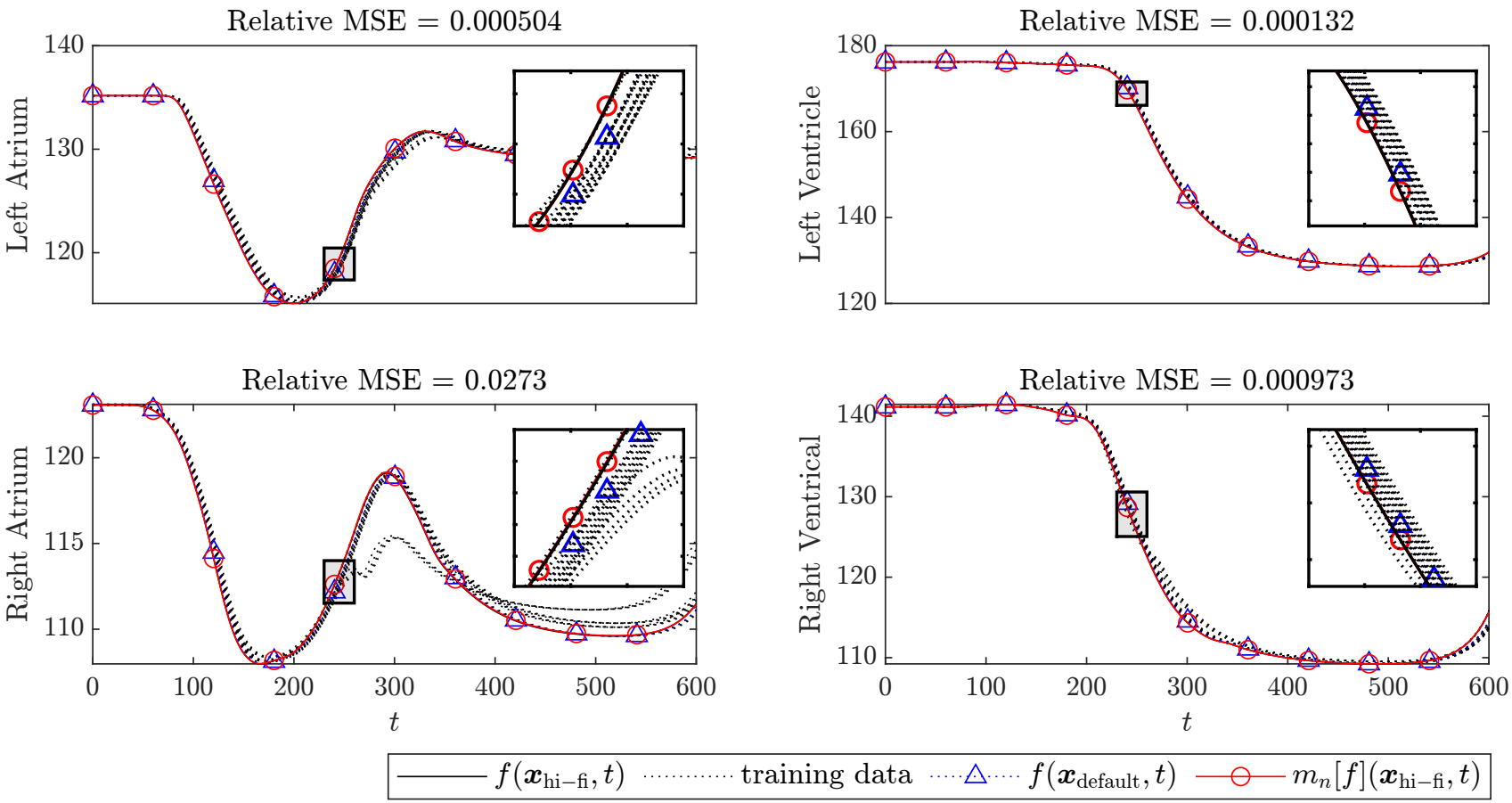
$$\frac{|f(\mathbf{x}_{\text{hi-fi}}) - m_n[f](\mathbf{x}_{\text{hi-fi}})|}{|f(\mathbf{x}_{\text{hi-fi}}) - f(\mathbf{x}_{\text{default}})|}$$

with respect to the default approximation, as a function of the total computational budget C.



Temporal quantities of interest

Extension of GRE to multi-output Gaussian process models allows temporal quantities of interest to be accurately extrapolated:



Summary

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\theta}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\theta}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\theta}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\theta}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\theta}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_\theta(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\boldsymbol{\theta}}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\boldsymbol{\theta}}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

Summary

- Black box probabilistic numerics leverages existing numerical methods in an extrapolation framework.
- Avoids the tricky issue of nonlinear information, by formulating a linear problem instead.
- Complementary to standard PN.

Full details in the report

`Probabilistic Richardson Extrapolation, arXiv:2401.07562`

where we discuss how Gauss–Richardson is capable also of

- estimation of uncertain convergence rates $b(\mathbf{x})$
- simultaneous extrapolation and emulation for parametric models $f_{\boldsymbol{\theta}}(\mathbf{x})$
- extrapolation of multivariate quantities $f(\mathbf{x}, \mathbf{t})$

Credit to: Onur Teymur, Chris Foley, Philip Breen, Toni Karvonen, Aretha Teckentrup, Marina Strocchi, Steve Niederer.

Thanks for listening!

References

L. F. Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Philosophical Transactions of the Royal Society of London, Series A*, 210(459-470):307–357, 1911.

M. Strocchi, S. Longobardi, C. M. Augustin, M. A. Gsell, A. Petras, C. A. Rinaldi, E. J. Vigmond, G. Plank, C. J. Oates, R. D. Wilkinson, et al. Cell to whole organ global sensitivity analysis on a four-chamber heart electromechanics model using Gaussian processes emulators. *PLOS Computational Biology*, 19(6):e1011257, 2023.

H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2004.