Black Box Probabilistic Numerics

Chris. J. Oates

September 2022
Gaussian Process Summer School

The Alan Turing Institute

Motivation

Conjugate Gaussian inference has been widely exploited in probabilistic numerics (PN), being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

Conjugate Gaussian inference has been widely exploited in PN, being the basis of methods developed for

- **linear algebra** [e.g. Cockayne et al., 2019, Bartels et al., 2019, Wenger and Hennig, 2020, Hennig, 2015, Reid et al., 2020, Schäfer et al., 2021, Bartels and Hennig, 2016, Cockayne et al., 2021]
- **cubature** [e.g. Diaconis, 1988, O'Hagan, 1992, Fisher et al., 2020, Prüher and Särkkä, 2016, Gessner et al., 2020, Karvonen et al., 2019, Chai and Garnett, 2019, Jagadeeswaran and Hickernell, 2019, Karvonen and Särkkä, 2017, Karvonen et al., 2018, Osborne et al., 2012, Xi et al., 2018, Briol et al., 2015, Gunter et al., 2014, Kennedy, 1998, O'Hagan, 1991, Larkin, 1972, Rasmussen and Ghahramani, 2003, Briol et al., 2019]
- **optimisation** [e.g. Mockus, 1977, Mockus et al., 1978, Mockus, 1989, Snoek et al., 2012, Hennig and Kiefel, 2013, Mahsereci and Hennig, 2015]
- **differential equations** [e.g. Skilling, 1992, Chkrebtii et al., 2016, Schober et al., 2019, Teymur et al., 2016, 2018, Hennig and Hauberg, 2013, Kersting and Hennig, 2016, Schober et al., 2014, Owhadi, 2015, Tronarp et al., 2019, Wang et al., 2018, Cockayne et al., 2017, Chkrebtii and Campbell, 2019, Owhadi and Scovel, 2019, Kersting et al., 2020, Wang et al., 2021, Bosch et al., 2021]

Principal limitations of current PN:

- **nonlinear** information pose a major technical challenge to this approach, due to the absence of explicit conditioning formulae, so the current scope of PN is limited.
- **lack of important functionalities**, such as adaptivity, numerical well-conditioning, efficient use of computational resource, etc.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Linear Information?

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

e.g. for numerical integration we might have

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

Linear information enables us to use a conjugate Gaussian framework:

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

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

3. Push the remaining uncertainty through Q.

What is Nonlinear Information?

Using the same notation, consider instead

$$Mx = b, \qquad x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d.$$

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

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

So it seems natural to let

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

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

This problem is not easily fixed.

What is Nonlinear Information?

Using the same notation, consider instead

$$Mx = b, \qquad x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d.$$

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

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

So it seems natural to let

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

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

This problem is not easily fixed.

What is Nonlinear Information?

Using the same notation, consider instead

$$Mx = b, \qquad x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d.$$

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

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

So it seems natural to let

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

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

This problem is not easily fixed.

What is Nonlinear Information?

Using the same notation, consider instead

$$Mx = b, \qquad x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d.$$

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

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

So it seems natural to let

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

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

This problem is not easily fixed.

What is Nonlinear Information?

Using the same notation, consider instead

$$Mx = b, \qquad x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d.$$

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

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

So it seems natural to let

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

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

This problem is not easily fixed.

What is Nonlinear Information?

Using the same notation, consider instead

$$Mx = b, \qquad x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d.$$

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

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

So it seems natural to let

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

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

This problem is not easily fixed.

Aim of the Talk

Aim: A pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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 pragmatic solution 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 Richardson extrapolation.

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

Compared to 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.

Richardson Extrapolation

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - q(h)}{0 - h^\alpha} = \frac{q(\gamma h) - q(h)}{(\gamma h)^\alpha - h^\alpha}$$

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - [q^* + Ch^\alpha + \mathcal{O}(h^{\alpha+1})]}{-h^\alpha} = \frac{[q^* + C(\gamma h)^\alpha + \mathcal{O}(h^{\alpha+1})] - [q^* + Ch^\alpha + \mathcal{O}(h^{\alpha+1})]}{(\gamma h)^\alpha - h^\alpha}$$

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - [q^* + Ch^\alpha + \mathcal{O}(h^{\alpha+1})]}{-h^\alpha} = \frac{[\cancel{q^*} + C(\gamma h)^\alpha + \mathcal{O}(h^{\alpha+1})] - [\cancel{q^*} + Ch^\alpha + \mathcal{O}(h^{\alpha+1})]}{(\gamma h)^\alpha - h^\alpha}$$

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - [q^* + Ch^\alpha + \mathcal{O}(h^{\alpha+1})]}{-h^\alpha} = \frac{[\cancel{q^*} + C(\gamma\cancel{h})^\alpha + \mathcal{O}(\cancel{h}^{\alpha+1})] - [\cancel{q^*} + C\cancel{h}^\alpha + \mathcal{O}(\cancel{h}^{\alpha+1})]}{(\gamma\cancel{h})^\alpha - \cancel{h}^\alpha}$$

Richardson Extrapolation

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\frac{y - [q^* + Ch^\alpha + \mathcal{O}(h^{\alpha+1})]}{-h^\alpha} = C + \mathcal{O}(h)$$

Consider the simplest setting

$$\underbrace{q(h)}_{\text{numerical method}} = \underbrace{q^*}_{\text{quantity of interest}} + \underbrace{Ch^\alpha + \mathcal{O}(h^{\alpha+1})}_{\text{error of the numerical method}}$$

where

- $C \in \mathbb{R}$ (may be unknown)
- $\alpha > 0$ (known, and called the *order* of the method)
- the *cost* of computing $q(h)$ increases as $h \downarrow 0$

Proposition

Fix $\gamma \in (0,1)$ and let $q_\gamma(h)$ denote the height at which a straight line drawn through the points $(h^\alpha, q(h))$ and $((\gamma h)^\alpha, q(\gamma h))$ intersects the vertical axis in $\mathbb{R}^2$. Then q_γ is a numerical method of order $\alpha + 1$.

Proof: Equation of a line:

$$\implies \qquad y = q^* + \mathcal{O}(h^{\alpha+1}),$$

as claimed. ■

Richardson Extrapolation, Ct'd

Could go further and fit an order $p-1$ polynomial to p distinct points

$$\{(h_i^\alpha, q(h_i))\}_{i=1}^p,$$

then extrapolate this to $h=0$, to give an estimate for the quantity of interest with error $\mathcal{O}(h^{\alpha+p})$.

Problem: Higher-order polynomial extrapolation can be unstable [Runge, 1901].

Proposed Solutions: Bulirsch and Stoer [1964] propose instead a rational function interpolant. This allows both greater expressiveness and robustness than polynomial interpolation [though not necessarily as efficiently; Press et al., 2007]. Other so-called *extrapolation methods* in numerical analysis; a comprehensive historical survey can be found in Joyce [1971].

Uncertainty Quantification: Our aim is to develop an extrapolation method that provides probabilistic uncertainty quantification, leading to what we call *Black Box Probabilistic Numerics* (BBPN in the sequel).

Richardson Extrapolation, Ct'd

Could go further and fit an order $p-1$ polynomial to p distinct points

$$\{(h_i^\alpha, q(h_i))\}_{i=1}^p,$$

then extrapolate this to $h=0$, to give an estimate for the quantity of interest with error $\mathcal{O}(h^{\alpha+p})$.

Problem: Higher-order polynomial extrapolation can be unstable [Runge, 1901].

Proposed Solutions: Bulirsch and Stoer [1964] propose instead a rational function interpolant. This allows both greater expressiveness and robustness than polynomial interpolation [though not necessarily as efficiently; Press et al., 2007]. Other so-called *extrapolation methods* in numerical analysis; a comprehensive historical survey can be found in Joyce [1971].

Uncertainty Quantification: Our aim is to develop an extrapolation method that provides probabilistic uncertainty quantification, leading to what we call *Black Box Probabilistic Numerics* (BBPN in the sequel).

Richardson Extrapolation, Ct'd

Could go further and fit an order $p-1$ polynomial to p distinct points

$$\{(h_i^\alpha, q(h_i))\}_{i=1}^p,$$

then extrapolate this to $h=0$, to give an estimate for the quantity of interest with error $\mathcal{O}(h^{\alpha+p})$.

Problem: Higher-order polynomial extrapolation can be unstable [Runge, 1901].

Proposed Solutions: Bulirsch and Stoer [1964] propose instead a rational function interpolant. This allows both greater expressiveness and robustness than polynomial interpolation [though not necessarily as efficiently; Press et al., 2007]. Other so-called *extrapolation methods* in numerical analysis; a comprehensive historical survey can be found in Joyce [1971].

Uncertainty Quantification: Our aim is to develop an extrapolation method that provides probabilistic uncertainty quantification, leading to what we call *Black Box Probabilistic Numerics* (BBPN in the sequel).

Richardson Extrapolation, Ct'd

Could go further and fit an order $p-1$ polynomial to p distinct points

$$\{(h_i^\alpha, q(h_i))\}_{i=1}^p,$$

then extrapolate this to $h=0$, to give an estimate for the quantity of interest with error $\mathcal{O}(h^{\alpha+p})$.

Problem: Higher-order polynomial extrapolation can be unstable [Runge, 1901].

Proposed Solutions: Bulirsch and Stoer [1964] propose instead a rational function interpolant. This allows both greater expressiveness and robustness than polynomial interpolation [though not necessarily as efficiently; Press et al., 2007]. Other so-called *extrapolation methods* in numerical analysis; a comprehensive historical survey can be found in Joyce [1971].

Uncertainty Quantification: Our aim is to develop an extrapolation method that provides probabilistic uncertainty quantification, leading to what we call *Black Box Probabilistic Numerics* (BBPN in the sequel).

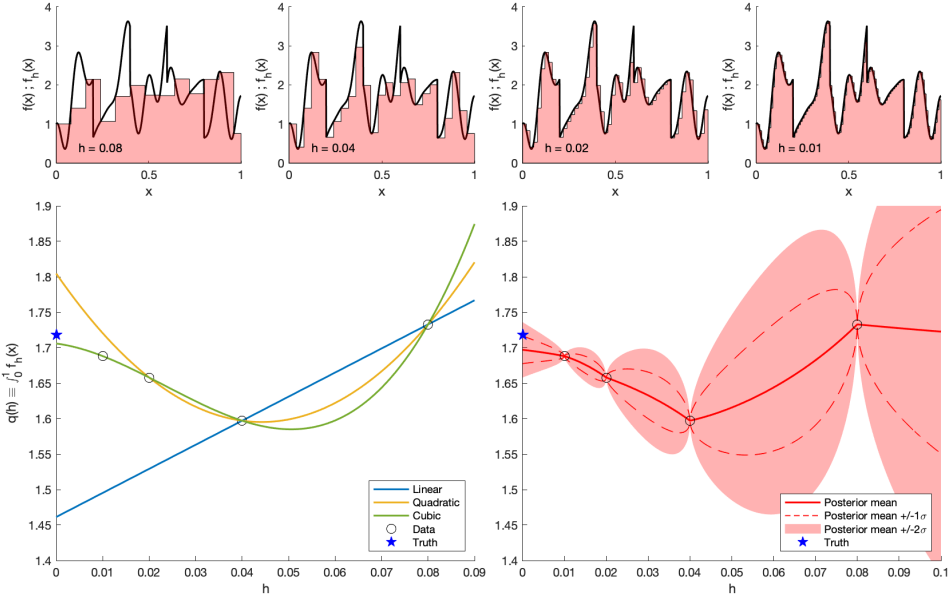

Figure: Richardson extrapolation for the Riemann sum method (left) and BBPN (right).

Methodology

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in black box probabilistic numerics (BBPN), since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in BBPN, since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in BBPN, since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in BBPN, since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in BBPN, since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in BBPN, since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

Notation and Setup

Idea: Model $q(h)$ as a *stochastic process* $Q(h)$, rather than fit a deterministic interpolant.

$\implies$ The distribution of $Q(0)$ is the epistemic uncertainty in the quantity of interest $q(0)$.

Linear: Conjugate Gaussian inference can be performed in BBPN, since one needs only to construct an interpolant.

Problem: How to formulate this in the abstract?

Definition (Traditional numerical method)

A traditional numerical method is defined as a map $q : [0, h_0) \times T \to \mathbb{R}$, for some $h_0 > 0$ such that, for all $t \in T$, the function $h \mapsto q(h, t)$ is continuous at 0 with limit $q(0, t) = q^*(t)$.

- the index t could be spatio-temporal, discrete, or even an unstructured set
- the index h will depend on the numerical method, e.g.
  - an error tolerance that is user-specified
  - $h = 1/\kappa$ with κ an iteration count

  but for simplicity we consider a single scalar h index.

BBPN begins with a *prior* stochastic process Q and constrains this prior using data

$$D := \{(h_i, t_{i,j}, q(h_i, t_{i,j})) : \ i = 1, \ldots, n; \ j = 1, \ldots, m_i\}$$

at resolutions $h_1 > \cdots > h_n > 0$ and distinct ordinates $t_{i,1}, \ldots, t_{i,m_i} \in T$.

The stochastic process obtained by conditioning Q on the dataset D, denoted $Q|D$, implies a marginal distribution for $Q(0, \cdot)$, which we interpret as a statistical prediction for the unknown quantity of interest $q^*(\cdot)$.

Meaningful uncertainty quantification (UQ) requires either

- expert knowledge about the numerical method that generated D, or
- a stochastic process that is able to adapt to the data, so that its predictions can be calibrated.

$\Longrightarrow$ Gaussian process (GP) models whose hyper-parameters are learned.

BBPN begins with a *prior* stochastic process Q and constrains this prior using data

$$D := \{(h_i, t_{i,j}, q(h_i, t_{i,j})) : \ i = 1, \ldots, n; \ j = 1, \ldots, m_i\}$$

at resolutions $h_1 > \cdots > h_n > 0$ and distinct ordinates $t_{i,1}, \ldots, t_{i,m_i} \in T$.

The stochastic process obtained by conditioning Q on the dataset D, denoted $Q|D$, implies a marginal distribution for $Q(0, \cdot)$, which we interpret as a statistical prediction for the unknown quantity of interest $q^*(\cdot)$.

Meaningful UQ requires either

- expert knowledge about the numerical method that generated D, or
- a stochastic process that is able to adapt to the data, so that its predictions can be calibrated.

$\Longrightarrow$ GP models whose hyper-parameters are learned.

BBPN begins with a *prior* stochastic process Q and constrains this prior using data

$$D := \{(h_i, t_{i,j}, q(h_i, t_{i,j})) : \ i = 1, \ldots, n; \ j = 1, \ldots, m_i\}$$

at resolutions $h_1 > \cdots > h_n > 0$ and distinct ordinates $t_{i,1}, \ldots, t_{i,m_i} \in T$.

The stochastic process obtained by conditioning Q on the dataset D, denoted $Q|D$, implies a marginal distribution for $Q(0, \cdot)$, which we interpret as a statistical prediction for the unknown quantity of interest $q^*(\cdot)$.

Meaningful UQ requires either

- expert knowledge about the numerical method that generated D, or
- a stochastic process that is able to adapt to the data, so that its predictions can be calibrated.

$\Longrightarrow$ GP models whose hyper-parameters are learned.

BBPN begins with a *prior* stochastic process Q and constrains this prior using data

$$D := \{(h_i, t_{i,j}, q(h_i, t_{i,j})) : \ i = 1, \ldots, n; \ j = 1, \ldots, m_i\}$$

at resolutions $h_1 > \cdots > h_n > 0$ and distinct ordinates $t_{i,1}, \ldots, t_{i,m_i} \in T$.

The stochastic process obtained by conditioning Q on the dataset D, denoted $Q|D$, implies a marginal distribution for $Q(0, \cdot)$, which we interpret as a statistical prediction for the unknown quantity of interest $q^*(\cdot)$.

Meaningful UQ requires either

- expert knowledge about the numerical method that generated D, or
- a stochastic process that is able to adapt to the data, so that its predictions can be calibrated.

$\Longrightarrow$ GP models whose hyper-parameters are learned.

BBPN begins with a *prior* stochastic process Q and constrains this prior using data

$$D := \{(h_i, t_{i,j}, q(h_i, t_{i,j})) : \ i = 1, \ldots, n; \ j = 1, \ldots, m_i\}$$

at resolutions $h_1 > \cdots > h_n > 0$ and distinct ordinates $t_{i,1}, \ldots, t_{i,m_i} \in T$.

The stochastic process obtained by conditioning Q on the dataset D, denoted $Q|D$, implies a marginal distribution for $Q(0, \cdot)$, which we interpret as a statistical prediction for the unknown quantity of interest $q^*(\cdot)$.

Meaningful UQ requires either

- expert knowledge about the numerical method that generated D, or
- a stochastic process that is able to adapt to the data, so that its predictions can be calibrated.

$\Longrightarrow$ GP models whose hyper-parameters are learned.

BBPN begins with a *prior* stochastic process Q and constrains this prior using data

$$D := \{(h_i, t_{i,j}, q(h_i, t_{i,j})) : \ i = 1, \ldots, n; \ j = 1, \ldots, m_i\}$$

at resolutions $h_1 > \cdots > h_n > 0$ and distinct ordinates $t_{i,1}, \ldots, t_{i,m_i} \in T$.

The stochastic process obtained by conditioning Q on the dataset D, denoted $Q|D$, implies a marginal distribution for $Q(0, \cdot)$, which we interpret as a statistical prediction for the unknown quantity of interest $q^*(\cdot)$.

Meaningful UQ requires either

- expert knowledge about the numerical method that generated D, or
- a stochastic process that is able to adapt to the data, so that its predictions can be calibrated.

$\Longrightarrow$ GP models whose hyper-parameters are learned.

Gaussian Process Model

Our goal is to specify a stochastic process model $Q(h,t)$ that behaves in a desirable way under extrapolation to $h = 0$.

To this end, we decompose

$$Q(h,t) = Q^*(t) + E(h,t)$$

where

- $Q^*(t)$ is a prior model for the unknown quantity of interest $q^*(t)$;
- $E(h,t)$ is a prior model for the error of the numerical method.

It will be assumed that Q^* and E are Gaussian and independent (written $Q^* \perp\!\!\!\perp E$, meaning that prior belief about the quantity of interest is independent of prior belief regarding the performance of the numerical method).

Compared to the aforementioned PN methods, a prior model for the error E is an additional requirement in BBPN.

Problem: The error $E(h,t)$ is known to vanish as $h \to 0$, meaning that a stationary GP model for $E(h,t)$, and hence for $Q(h,t)$, is inappropriate for UQ.

Gaussian Process Model

Our goal is to specify a stochastic process model $Q(h,t)$ that behaves in a desirable way under extrapolation to $h = 0$.

To this end, we decompose

$$Q(h,t) = Q^*(t) + E(h,t)$$

where

- $Q^*(t)$ is a prior model for the unknown quantity of interest $q^*(t)$;
- $E(h,t)$ is a prior model for the error of the numerical method.

It will be assumed that Q^* and E are Gaussian and independent (written $Q^* \perp\!\!\!\perp E$, meaning that prior belief about the quantity of interest is independent of prior belief regarding the performance of the numerical method).

Compared to the aforementioned PN methods, a prior model for the error E is an additional requirement in BBPN.

Problem: The error $E(h,t)$ is known to vanish as $h \to 0$, meaning that a stationary GP model for $E(h,t)$, and hence for $Q(h,t)$, is inappropriate for UQ.

Gaussian Process Model

Our goal is to specify a stochastic process model $Q(h,t)$ that behaves in a desirable way under extrapolation to $h = 0$.

To this end, we decompose

$$Q(h,t) = Q^*(t) + E(h,t)$$

where

- $Q^*(t)$ is a prior model for the unknown quantity of interest $q^*(t)$;
- $E(h,t)$ is a prior model for the error of the numerical method.

It will be assumed that Q^* and E are Gaussian and independent (written $Q^* \perp\!\!\!\perp E$, meaning that prior belief about the quantity of interest is independent of prior belief regarding the performance of the numerical method).

Compared to the aforementioned PN methods, a prior model for the error E is an additional requirement in BBPN.

Problem: The error $E(h,t)$ is known to vanish as $h \to 0$, meaning that a stationary GP model for $E(h,t)$, and hence for $Q(h,t)$, is inappropriate for UQ.

Gaussian Process Model

Our goal is to specify a stochastic process model $Q(h,t)$ that behaves in a desirable way under extrapolation to $h = 0$.

To this end, we decompose

$$Q(h,t) = Q^*(t) + E(h,t)$$

where

- $Q^*(t)$ is a prior model for the unknown quantity of interest $q^*(t)$;
- $E(h,t)$ is a prior model for the error of the numerical method.

It will be assumed that Q^* and E are Gaussian and independent (written $Q^* \perp\!\!\!\perp E$, meaning that prior belief about the quantity of interest is independent of prior belief regarding the performance of the numerical method).

Compared to the aforementioned PN methods, a prior model for the error E is an additional requirement in BBPN.

Problem: The error $E(h,t)$ is known to vanish as $h \to 0$, meaning that a stationary GP model for $E(h,t)$, and hence for $Q(h,t)$, is inappropriate for UQ.

Gaussian Process Model

Our goal is to specify a stochastic process model $Q(h,t)$ that behaves in a desirable way under extrapolation to $h = 0$.

To this end, we decompose

$$Q(h,t) = Q^*(t) + E(h,t)$$

where

- $Q^*(t)$ is a prior model for the unknown quantity of interest $q^*(t)$;
- $E(h,t)$ is a prior model for the error of the numerical method.

It will be assumed that Q^* and E are Gaussian and independent (written $Q^* \perp\!\!\!\perp E$, meaning that prior belief about the quantity of interest is independent of prior belief regarding the performance of the numerical method).

Compared to the aforementioned PN methods, a prior model for the error E is an additional requirement in BBPN.

Problem: The error $E(h,t)$ is known to vanish as $h \to 0$, meaning that a stationary GP model for $E(h,t)$, and hence for $Q(h,t)$, is inappropriate for UQ.

Prior for Q^*: In the absence of detailed prior belief about q^*, we consider the following default prior model:

$$Q^*(t) = Z \cdot b(t) + G(t), \qquad b(t) = (b_1(t), \ldots, b_v(t))^\top$$
$$Z = (Z_1, \ldots, Z_v)^\top \sim \mathcal{N}(0, \sigma^2 I), \qquad Z \perp\!\!\!\perp G$$
$$G \sim \mathcal{GP}(0, \sigma^2 \rho_G k_G),$$

- $\sigma^2, \rho_G > 0$ control the UQ and are to be estimated
- the basis b will be problem-specific and could be a polynomial basis, Fourier basis, or any number of other bases depending on context.
- The case $v = 1$ with a constant intercept is closely related to *ordinary kriging* and the case $v > 1$ is closely related to *universal kriging* [Stein, 2012, p. 8].
- Using the notation $t = (t_1, \ldots, t_p)$, we consider a tensor product covariance model $k_G(t, t') = \prod_{i=1}^p k_{G,i}(t_i, t_i')$, $k_{G,i}(t_i, t_i') = \phi_i(\|t_i - t_i'\|/\ell_{t,i})$, for some radial basis functions ϕ_i, scaled to satisfy $\phi_i(0) = 1$, and additional length-scale parameters $\ell_{t,i} > 0$ to be estimated.

Prior for Q^*: In the absence of detailed prior belief about q^*, we consider the following default prior model:

$$Q^*(t) = Z \cdot b(t) + G(t), \qquad b(t) = (b_1(t), \ldots, b_v(t))^\top$$
$$Z = (Z_1, \ldots, Z_v)^\top \sim \mathcal{N}(0, \sigma^2 I), \qquad Z \perp\!\!\!\perp G$$
$$G \sim \mathcal{GP}(0, \sigma^2 \rho_G k_G),$$

- $\sigma^2, \rho_G > 0$ control the UQ and are to be estimated
- the basis b will be problem-specific and could be a polynomial basis, Fourier basis, or any number of other bases depending on context.
- The case $v = 1$ with a constant intercept is closely related to *ordinary kriging* and the case $v > 1$ is closely related to *universal kriging* [Stein, 2012, p. 8].
- Using the notation $t = (t_1, \ldots, t_p)$, we consider a tensor product covariance model $k_G(t, t') = \prod_{i=1}^p k_{G,i}(t_i, t_i')$, $k_{G,i}(t_i, t_i') = \phi_i(\|t_i - t_i'\|/\ell_{t,i})$, for some radial basis functions ϕ_i, scaled to satisfy $\phi_i(0) = 1$, and additional length-scale parameters $\ell_{t,i} > 0$ to be estimated.

Prior for Q^*: In the absence of detailed prior belief about q^*, we consider the following default prior model:

$$Q^*(t) = Z \cdot b(t) + G(t), \qquad b(t) = (b_1(t), \ldots, b_v(t))^\top$$
$$Z = (Z_1, \ldots, Z_v)^\top \sim \mathcal{N}(0, \sigma^2 I), \qquad Z \perp\!\!\!\perp G$$
$$G \sim \mathcal{GP}(0, \sigma^2 \rho_G k_G),$$

- $\sigma^2, \rho_G > 0$ control the UQ and are to be estimated
- the basis b will be problem-specific and could be a polynomial basis, Fourier basis, or any number of other bases depending on context.
- The case $v = 1$ with a constant intercept is closely related to *ordinary kriging* and the case $v > 1$ is closely related to *universal kriging* [Stein, 2012, p. 8].
- Using the notation $t = (t_1, \ldots, t_p)$, we consider a tensor product covariance model $k_G(t, t') = \prod_{i=1}^p k_{G,i}(t_i, t_i')$, $k_{G,i}(t_i, t_i') = \phi_i(\|t_i - t_i'\|/\ell_{t,i})$, for some radial basis functions ϕ_i, scaled to satisfy $\phi_i(0) = 1$, and additional length-scale parameters $\ell_{t,i} > 0$ to be estimated.

Prior for Q^*: In the absence of detailed prior belief about q^*, we consider the following default prior model:

$$Q^*(t) = Z \cdot b(t) + G(t), \qquad b(t) = (b_1(t), \ldots, b_v(t))^\top$$
$$Z = (Z_1, \ldots, Z_v)^\top \sim \mathcal{N}(0, \sigma^2 I), \qquad Z \perp\!\!\!\perp G$$
$$G \sim \mathcal{GP}(0, \sigma^2 \rho_G k_G),$$

- $\sigma^2, \rho_G > 0$ control the UQ and are to be estimated
- the basis b will be problem-specific and could be a polynomial basis, Fourier basis, or any number of other bases depending on context.
- The case $v = 1$ with a constant intercept is closely related to *ordinary kriging* and the case $v > 1$ is closely related to *universal kriging* [Stein, 2012, p. 8].
- Using the notation $t = (t_1, \ldots, t_p)$, we consider a tensor product covariance model $k_G(t, t') = \prod_{i=1}^p k_{G,i}(t_i, t_i')$, $k_{G,i}(t_i, t_i') = \phi_i(\|t_i - t_i'\|/\ell_{t,i})$, for some radial basis functions ϕ_i, scaled to satisfy $\phi_i(0) = 1$, and additional length-scale parameters $\ell_{t,i} > 0$ to be estimated.

Prior for Q^*: In the absence of detailed prior belief about q^*, we consider the following default prior model:

$$Q^*(t) = Z \cdot b(t) + G(t), \qquad b(t) = (b_1(t), \ldots, b_v(t))^\top$$
$$Z = (Z_1, \ldots, Z_v)^\top \sim \mathcal{N}(0, \sigma^2 I), \qquad Z \perp\!\!\!\perp G$$
$$G \sim \mathcal{GP}(0, \sigma^2 \rho_G k_G),$$

- $\sigma^2, \rho_G > 0$ control the UQ and are to be estimated
- the basis b will be problem-specific and could be a polynomial basis, Fourier basis, or any number of other bases depending on context.
- The case $v = 1$ with a constant intercept is closely related to *ordinary kriging* and the case $v > 1$ is closely related to *universal kriging* [Stein, 2012, p. 8].
- Using the notation $t = (t_1, \ldots, t_p)$, we consider a tensor product covariance model $k_G(t, t') = \prod_{i=1}^p k_{G,i}(t_i, t_i')$, $k_{G,i}(t_i, t_i') = \phi_i(\|t_i - t_i'\|/\ell_{t,i})$, for some radial basis functions ϕ_i, scaled to satisfy $\phi_i(0) = 1$, and additional length-scale parameters $\ell_{t,i} > 0$ to be estimated.

Prior for E: The process $E(h,t)$ is a model for the numerical error $q(h,t) - q^*(t)$, $t > 0$, which may be highly structured. A flexible prior model is therefore required, and here is our default:

$$E(h,t) = h^\alpha \tilde{E}(h,t),$$
$$\tilde{E} \sim \mathcal{GP}(0, \sigma^2 \rho_E k_{\tilde{E}}),$$

- $\rho_E > 0$ is a parameter to be estimated
- $k_{\tilde{E}}(h,t) = \psi(|h-h'|/\ell_h) \cdot k_G(t,t')$, for a radial basis function ψ, scaled to satisfy $\psi(0) = 1$, and a length-scale parameter $\ell_h > 0$ to be estimated.
- $\implies k_E((h,t),(h',t')) = (hh')^\alpha k_{\tilde{E}}(h,t)$

Overall prior for Q: Our default model can be interpreted as universal kriging over T, with a covariance adjusted by a multiplicative error arising from non-zero values of h:

$$k_Q((h,t),(h',t')) = \sigma^2 \left\{ b(t) \cdot b(t') + \rho_G k_G(t,t') \left(1 + \rho_E \frac{k_E((h,t),(h',t'))}{k_G(t,t')} \right) \right\},$$

where k_E/k_G is a kernel only depending on h.

Prior for E: The process $E(h,t)$ is a model for the numerical error $q(h,t) - q^*(t)$, $t > 0$, which may be highly structured. A flexible prior model is therefore required, and here is our default:

$$E(h,t) = h^\alpha \tilde{E}(h,t),$$
$$\tilde{E} \sim \mathcal{GP}(0, \sigma^2 \rho_E k_{\tilde{E}}),$$

- $\rho_E > 0$ is a parameter to be estimated
- $k_{\tilde{E}}(h,t) = \psi(|h - h'|/\ell_h) \cdot k_G(t,t')$, for a radial basis function ψ, scaled to satisfy $\psi(0) = 1$, and a length-scale parameter $\ell_h > 0$ to be estimated.
- $\implies k_E((h,t),(h',t')) = (hh')^\alpha k_{\tilde{E}}(h,t)$

Overall prior for Q: Our default model can be interpreted as universal kriging over T, with a covariance adjusted by a multiplicative error arising from non-zero values of h:

$$k_Q((h,t),(h',t')) = \sigma^2 \left\{ b(t) \cdot b(t') + \rho_G k_G(t,t') \left(1 + \rho_E \frac{k_E((h,t),(h',t'))}{k_G(t,t')} \right) \right\},$$

where k_E/k_G is a kernel only depending on h.

Gaussian Process Model

Prior for E: The process $E(h,t)$ is a model for the numerical error $q(h,t) - q^*(t)$, $t > 0$, which may be highly structured. A flexible prior model is therefore required, and here is our default:

$$E(h,t) = h^\alpha \tilde{E}(h,t),$$
$$\tilde{E} \sim \mathcal{GP}(0, \sigma^2 \rho_E k_{\tilde{E}}),$$

- $\rho_E > 0$ is a parameter to be estimated
- $k_{\tilde{E}}(h,t) = \psi(|h - h'|/\ell_h) \cdot k_G(t,t')$, for a radial basis function ψ, scaled to satisfy $\psi(0) = 1$, and a length-scale parameter $\ell_h > 0$ to be estimated.
- $\implies k_E((h,t),(h',t')) = (hh')^\alpha k_{\tilde{E}}(h,t)$

Overall prior for Q: Our default model can be interpreted as universal kriging over T, with a covariance adjusted by a multiplicative error arising from non-zero values of h:

$$k_Q((h,t),(h',t')) = \sigma^2 \left\{ b(t) \cdot b(t') + \rho_G k_G(t,t') \left(1 + \rho_E \frac{k_E((h,t),(h',t'))}{k_G(t,t')} \right) \right\},$$

where k_E / k_G is a kernel only depending on h.

Prior for E: The process $E(h,t)$ is a model for the numerical error $q(h,t) - q^*(t)$, $t > 0$, which may be highly structured. A flexible prior model is therefore required, and here is our default:

$$E(h,t) = h^\alpha \tilde{E}(h,t),$$
$$\tilde{E} \sim \mathcal{GP}(0, \sigma^2 \rho_E k_{\tilde{E}}),$$

- $\rho_E > 0$ is a parameter to be estimated
- $k_{\tilde{E}}(h,t) = \psi(|h - h'|/\ell_h) \cdot k_G(t,t')$, for a radial basis function ψ, scaled to satisfy $\psi(0) = 1$, and a length-scale parameter $\ell_h > 0$ to be estimated.
- $\implies k_E((h,t),(h',t')) = (hh')^\alpha k_{\tilde{E}}(h,t)$

Overall prior for Q: Our default model can be interpreted as universal kriging over T, with a covariance adjusted by a multiplicative error arising from non-zero values of h:

$$k_Q((h,t),(h',t')) = \sigma^2 \left\{ b(t) \cdot b(t') + \rho_G k_G(t,t') \left(1 + \rho_E \frac{k_E((h,t),(h',t'))}{k_G(t,t')} \right) \right\},$$

where k_E / k_G is a kernel only depending on h.

Prior for E: The process $E(h,t)$ is a model for the numerical error $q(h,t) - q^*(t)$, $t > 0$, which may be highly structured. A flexible prior model is therefore required, and here is our default:

$$E(h,t) = h^\alpha \tilde{E}(h,t),$$
$$\tilde{E} \sim \mathcal{GP}(0, \sigma^2 \rho_E k_{\tilde{E}}),$$

- $\rho_E > 0$ is a parameter to be estimated
- $k_{\tilde{E}}(h,t) = \psi(|h - h'|/\ell_h) \cdot k_G(t,t')$, for a radial basis function ψ, scaled to satisfy $\psi(0) = 1$, and a length-scale parameter $\ell_h > 0$ to be estimated.
- $\implies k_E((h,t),(h',t')) = (hh')^\alpha k_{\tilde{E}}(h,t)$

Overall prior for Q: Our default model can be interpreted as universal kriging over T, with a covariance adjusted by a multiplicative error arising from non-zero values of h:

$$k_Q((h,t),(h',t')) = \sigma^2 \left\{ b(t) \cdot b(t') + \rho_G k_G(t,t') \left(1 + \rho_E \frac{k_E((h,t),(h',t'))}{k_G(t,t')} \right) \right\},$$

where k_E / k_G is a kernel only depending on h.

BBPN as an Extrapolation Method

The GP specification just described is not arbitrary; it ensures that the higher-order convergence property of Richardson extrapolation (RE) is realised in BBPN.

Proposition

(In the same setting as the earlier Proposition.) Suppose ψ is Lipschitz[1]. Then

$$|q^* - \mathbb{E}[Q(0)|D_h]| = \mathcal{O}(h^{\alpha+1})$$

as $h \to 0$.

- Can be extended to higher orders, like classical RE.
- Generalisation to general $t \in T$ still a work in progress (thoughts welcome!)

[1] e.g. Matérn covariance function of smoothness at least 1/2 is Lipschitz.

BBPN as an Extrapolation Method

The GP specification just described is not arbitrary; it ensures that the higher-order convergence property of RE is realised in BBPN.

Proposition

(In the same setting as the earlier Proposition.) Suppose ψ is Lipschitz[1]*. Then*

$$|q^* - \mathbb{E}[Q(0)|D_h]| = \mathcal{O}(h^{\alpha+1})$$

as $h \to 0$.

- Can be extended to higher orders, like classical RE.
- Generalisation to general $t \in T$ still a work in progress (thoughts welcome!)

[1] e.g. Matérn covariance function of smoothness at least $1/2$ is Lipschitz.

BBPN as an Extrapolation Method

The GP specification just described is not arbitrary; it ensures that the higher-order convergence property of RE is realised in BBPN.

Proposition

(In the same setting as the earlier Proposition.) Suppose ψ is Lipschitz[1]*. Then*

$$|q^* - \mathbb{E}[Q(0)|D_h]| = \mathcal{O}(h^{\alpha+1})$$

as $h \to 0$.

- Can be extended to higher orders, like classical RE.
- Generalisation to general $t \in T$ still a work in progress (thoughts welcome!)

[1] e.g. Matérn covariance function of smoothness at least $1/2$ is Lipschitz.

BBPN as an Extrapolation Method

The GP specification just described is not arbitrary; it ensures that the higher-order convergence property of RE is realised in BBPN.

Proposition

(In the same setting as the earlier Proposition.) Suppose ψ is Lipschitz[1]*. Then*

$$|q^* - \mathbb{E}[Q(0)|D_h]| = \mathcal{O}(h^{\alpha+1})$$

as $h \to 0$.

- Can be extended to higher orders, like classical RE.
- Generalisation to general $t \in T$ still a work in progress (thoughts welcome!)

[1] e.g. Matérn covariance function of smoothness at least $1/2$ is Lipschitz.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

The free parameters of our GP model are

$$\theta = \{\sigma^2, \rho_G, \rho_E, \ell_h, \ell_{t,i},\ i = 1, \ldots, p\}$$

and will be set using the maximum likelihood:

(✓) no degrees of freedom (such as the number of folds of cross-validation) permits a more objective empirical assessment.

(✓) σ^2_{ML} has a closed form expression in terms of the remaining parameters.

(✓) gradients with respect to the remaining $3 + p$ parameters can be derived and exploited.

(✗) optimisation can be difficult when $p \gg 1$.

(✗) over-confident UQ at finite sample sizes [Karvonen and Oates, 2022].

Further remarks:

- GP interpolation, as with classical RE, is not parameterisation invariant.
- RE presupposes that the order α must be known *a priori*. However if α is not known, the probabilistic perspective affords us the opportunity to *learn* α as an additional parameter in the statistical model—a procedure with no classical analogue.

Experimental Assessment

Set-Up

Kernels: Matérn(1/2) kernels were used for ϕ_i and ψ, imposing a minimal continuity assumption on q without additional levels of smoothness being assumed.

Performance Metrics: The *error* of the point estimate (mean), is denoted

$$W := \|\mathbb{E}[Q(0,\cdot)|D] - q^*(\cdot)\|,$$

where the norm is taken over $t \in T'$ where T' is either T itself or a set of representative elements from T.

The *surprise* is denoted

$$S^2 := \|\mathbb{C}[Q(0,\cdot)|D]^{-1/2}(\mathbb{E}[Q(0,\cdot)|D)] - q^*(\cdot))\|,$$

where $\mathbb{C}[Q(0,\cdot)|D]$ denotes the posterior covariance matrix.

Note that if $q^* \sim Q(0,\cdot)|D$, then $S^2 \sim \chi^2_{|T'|}$, enabling calibration of UQ to be assessed.

Set-Up

Kernels: Matérn(1/2) kernels were used for ϕ_i and ψ, imposing a minimal continuity assumption on q without additional levels of smoothness being assumed.

Performance Metrics: The *error* of the point estimate (mean), is denoted

$$W \coloneqq \|\mathbb{E}[Q(0,\cdot)|D] - q^*(\cdot)\|,$$

where the norm is taken over $t \in T'$ where T' is either T itself or a set of representative elements from T.

The *surprise* is denoted

$$S^2 \coloneqq \|\mathbb{C}[Q(0,\cdot)|D]^{-1/2}(\mathbb{E}[Q(0,\cdot)|D)] - q^*(\cdot))\|,$$

where $\mathbb{C}[Q(0,\cdot)|D]$ denotes the posterior covariance matrix.

Note that if $q^* \sim Q(0,\cdot)|D$, then $S^2 \sim \chi^2_{|T'|}$, enabling calibration of UQ to be assessed.

Set-Up

Kernels: Matérn(1/2) kernels were used for ϕ_i and ψ, imposing a minimal continuity assumption on q without additional levels of smoothness being assumed.

Performance Metrics: The *error* of the point estimate (mean), is denoted

$$W \coloneqq \|\mathbb{E}[Q(0,\cdot)|D] - q^*(\cdot)\|,$$

where the norm is taken over $t \in T'$ where T' is either T itself or a set of representative elements from T.

The *surprise* is denoted

$$S^2 \coloneqq \|\mathbb{C}[Q(0,\cdot)|D]^{-1/2}(\mathbb{E}[Q(0,\cdot)|D)] - q^*(\cdot))\|,$$

where $\mathbb{C}[Q(0,\cdot)|D]$ denotes the posterior covariance matrix.

Note that if $q^* \sim Q(0,\cdot)|D$, then $S^2 \sim \chi^2_{|T'|}$, enabling calibration of UQ to be assessed.

Kernels: Matérn(1/2) kernels were used for ϕ_i and ψ, imposing a minimal continuity assumption on q without additional levels of smoothness being assumed.

Performance Metrics: The *error* of the point estimate (mean), is denoted

$$W \coloneqq \|\mathbb{E}[Q(0,\cdot)|D] - q^*(\cdot)\|,$$

where the norm is taken over $t \in T'$ where T' is either T itself or a set of representative elements from T.

The *surprise* is denoted

$$S^2 \coloneqq \|\mathbb{C}[Q(0,\cdot)|D]^{-1/2}(\mathbb{E}[Q(0,\cdot)|D)] - q^*(\cdot))\|,$$

where $\mathbb{C}[Q(0,\cdot)|D]$ denotes the posterior covariance matrix.

Note that if $q^* \sim Q(0,\cdot)|D$, then $S^2 \sim \chi^2_{|T'|}$, enabling calibration of UQ to be assessed.

Ordinary Differential Equations

Consider the following Lotka–Volterra initial value problem (IVP), a popular test case in PN:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = f(t, \mathbf{y}) = \begin{bmatrix} 0.5y_1 - 0.05y_1y_2 \\ -0.5y_2 + 0.05y_1y_2 \end{bmatrix}, \quad \mathbf{y}(0) = \begin{bmatrix} 20 \\ 20 \end{bmatrix}$$

The aim in what follows is to approximate the quantity of interest $q^* = \mathbf{y}(t_{\text{end}})$ for $t_{\text{end}} = 20$.

Methods considered: Chkr. [Chkrebtii et al., 2016]; Conr. O1 [Conrad et al., 2016]; Teym. O2 [Teymur et al., 2016]; Scho. O1 [Schober et al., 2019]; Tron. O2 [Tronarp et al., 2019]; Bosch O2 [Bosch et al., 2021]; and BBPN O1 & O2.

The differing character of existing PN methods makes direct comparisons challenging, particularly if we are to account for computational cost:

- Chkr., Conr. O1, and Teym. O2 require parallel simulations to produce empirical credible sets, and thus have a significant computational cost.
- Scho. O1, Tron. O2, and Bosch O2 are based on Gaussian filtering and are less computationally demanding, though provide less expressive UQ.
- BBPN used the final states produced by either Euler (O1) or Adams–Bashforth (O2), at resolutions $h_i = 2^{-i}$.

However, each algorithm has a recognisable discretisation parameter h, so it remains instructive to study their $h \to 0$ limit. (For BBPN, h is the finest resolution considered.)

Ordinary Differential Equations

Consider the following Lotka–Volterra IVP, a popular test case in PN:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = f(t, \mathbf{y}) = \begin{bmatrix} 0.5y_1 - 0.05y_1y_2 \\ -0.5y_2 + 0.05y_1y_2 \end{bmatrix}, \quad \mathbf{y}(0) = \begin{bmatrix} 20 \\ 20 \end{bmatrix}$$

The aim in what follows is to approximate the quantity of interest $q^* = \mathbf{y}(t_{\text{end}})$ for $t_{\text{end}} = 20$.

Methods considered: Chkr. [Chkrebtii et al., 2016]; Conr. O1 [Conrad et al., 2016]; Teym. O2 [Teymur et al., 2016]; Scho. O1 [Schober et al., 2019]; Tron. O2 [Tronarp et al., 2019]; Bosch O2 [Bosch et al., 2021]; and BBPN O1 & O2.

The differing character of existing PN methods makes direct comparisons challenging, particularly if we are to account for computational cost:

- Chkr., Conr. O1, and Teym. O2 require parallel simulations to produce empirical credible sets, and thus have a significant computational cost.
- Scho. O1, Tron. O2, and Bosch O2 are based on Gaussian filtering and are less computationally demanding, though provide less expressive UQ.
- BBPN used the final states produced by either Euler (O1) or Adams–Bashforth (O2), at resolutions $h_i = 2^{-i}$.

However, each algorithm has a recognisable discretisation parameter h, so it remains instructive to study their $h \to 0$ limit. (For BBPN, h is the finest resolution considered.)

Ordinary Differential Equations

Consider the following Lotka–Volterra IVP, a popular test case in PN:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = f(t, \mathbf{y}) = \begin{bmatrix} 0.5y_1 - 0.05y_1y_2 \\ -0.5y_2 + 0.05y_1y_2 \end{bmatrix}, \quad \mathbf{y}(0) = \begin{bmatrix} 20 \\ 20 \end{bmatrix}$$

The aim in what follows is to approximate the quantity of interest $q^* = \mathbf{y}(t_{\text{end}})$ for $t_{\text{end}} = 20$.

Methods considered: Chkr. [Chkrebtii et al., 2016]; Conr. O1 [Conrad et al., 2016]; Teym. O2 [Teymur et al., 2016]; Scho. O1 [Schober et al., 2019]; Tron. O2 [Tronarp et al., 2019]; Bosch O2 [Bosch et al., 2021]; and BBPN O1 & O2.

The differing character of existing PN methods makes direct comparisons challenging, particularly if we are to account for computational cost:

- Chkr., Conr. O1, and Teym. O2 require parallel simulations to produce empirical credible sets, and thus have a significant computational cost.
- Scho. O1, Tron. O2, and Bosch O2 are based on Gaussian filtering and are less computationally demanding, though provide less expressive UQ.
- BBPN used the final states produced by either Euler (O1) or Adams–Bashforth (O2), at resolutions $h_i = 2^{-i}$.

However, each algorithm has a recognisable discretisation parameter h, so it remains instructive to study their $h \to 0$ limit. (For BBPN, h is the finest resolution considered.)

Ordinary Differential Equations

Consider the following Lotka–Volterra IVP, a popular test case in PN:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = f(t, \mathbf{y}) = \begin{bmatrix} 0.5y_1 - 0.05y_1y_2 \\ -0.5y_2 + 0.05y_1y_2 \end{bmatrix}, \quad \mathbf{y}(0) = \begin{bmatrix} 20 \\ 20 \end{bmatrix}$$

The aim in what follows is to approximate the quantity of interest $q^* = \mathbf{y}(t_{\text{end}})$ for $t_{\text{end}} = 20$.

Methods considered: Chkr. [Chkrebtii et al., 2016]; Conr. O1 [Conrad et al., 2016]; Teym. O2 [Teymur et al., 2016]; Scho. O1 [Schober et al., 2019]; Tron. O2 [Tronarp et al., 2019]; Bosch O2 [Bosch et al., 2021]; and BBPN O1 & O2.

The differing character of existing PN methods makes direct comparisons challenging, particularly if we are to account for computational cost:

- Chkr., Conr. O1, and Teym. O2 require parallel simulations to produce empirical credible sets, and thus have a significant computational cost.
- Scho. O1, Tron. O2, and Bosch O2 are based on Gaussian filtering and are less computationally demanding, though provide less expressive UQ.
- BBPN used the final states produced by either Euler (O1) or Adams–Bashforth (O2), at resolutions $h_i = 2^{-i}$.

However, each algorithm has a recognisable discretisation parameter h, so it remains instructive to study their $h \to 0$ limit. (For BBPN, h is the finest resolution considered.)

Ordinary Differential Equations

Consider the following Lotka–Volterra IVP, a popular test case in PN:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = f(t, \mathbf{y}) = \begin{bmatrix} 0.5y_1 - 0.05y_1y_2 \\ -0.5y_2 + 0.05y_1y_2 \end{bmatrix}, \quad \mathbf{y}(0) = \begin{bmatrix} 20 \\ 20 \end{bmatrix}$$

The aim in what follows is to approximate the quantity of interest $q^* = \mathbf{y}(t_{\text{end}})$ for $t_{\text{end}} = 20$.

Methods considered: Chkr. [Chkrebtii et al., 2016]; Conr. O1 [Conrad et al., 2016]; Teym. O2 [Teymur et al., 2016]; Scho. O1 [Schober et al., 2019]; Tron. O2 [Tronarp et al., 2019]; Bosch O2 [Bosch et al., 2021]; and BBPN O1 & O2.

The differing character of existing PN methods makes direct comparisons challenging, particularly if we are to account for computational cost:

- Chkr., Conr. O1, and Teym. O2 require parallel simulations to produce empirical credible sets, and thus have a significant computational cost.
- Scho. O1, Tron. O2, and Bosch O2 are based on Gaussian filtering and are less computationally demanding, though provide less expressive UQ.
- BBPN used the final states produced by either Euler (O1) or Adams–Bashforth (O2), at resolutions $h_i = 2^{-i}$.

However, each algorithm has a recognisable discretisation parameter h, so it remains instructive to study their $h \to 0$ limit. (For BBPN, h is the finest resolution considered.)

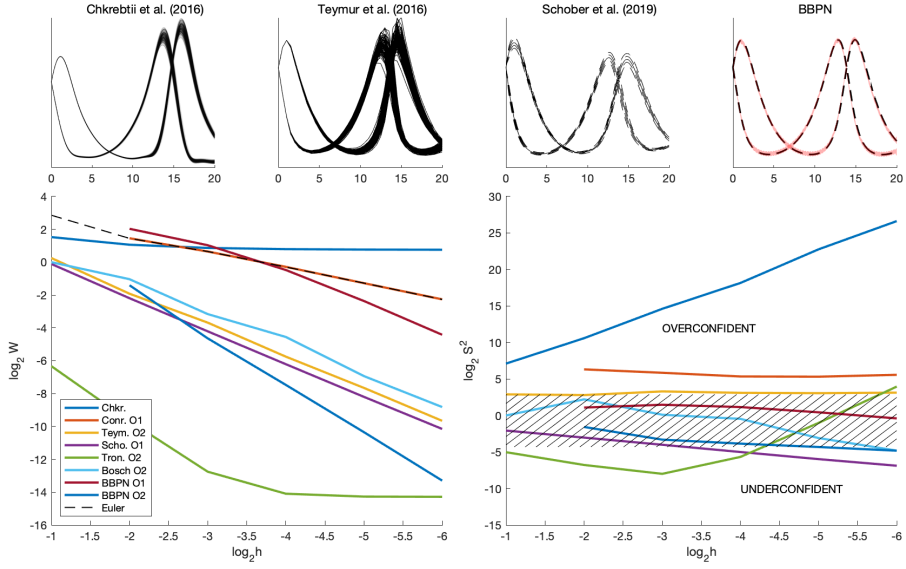

Figure: Top: Output from three existing PN algorithms Chkrebtii et al. [2016], Teymur et al. [2016], Schober et al. [2019] and BBPN. Bottom left: The error $\log_2 W$ at the final time point $t_{\mathrm{end}} = 20$, as a function of the time step size h. Bottom right: The surprise $\log_2 S$ at $t_{\mathrm{end}} = 20$, with the central 95% probability band of a χ^2_2 random variable shaded.

In this experiment:

- (✓) BBPN is observed to be calibrated.
- (✓) BBPN (O2) provides the most accurate approximation among all methods that are calibrated.
- (✓) BBPN accelerates the convergence of the Euler method from first order to second order, akin to RE.
- (∼) The computational cost of BBPN was intermediate between the filtering approach of Schober et al. [2019] and the sampling approaches of Chkrebtii et al. [2016] and Teymur et al. [2016].

In this experiment:

(✓) BBPN is observed to be calibrated.

(✓) BBPN (O2) provides the most accurate approximation among all methods that are calibrated.

(✓) BBPN accelerates the convergence of the Euler method from first order to second order, akin to RE.

(∼) The computational cost of BBPN was intermediate between the filtering approach of Schober et al. [2019] and the sampling approaches of Chkrebtii et al. [2016] and Teymur et al. [2016].

In this experiment:

- (✓) BBPN is observed to be calibrated.
- (✓) BBPN (O2) provides the most accurate approximation among all methods that are calibrated.
- (✓) BBPN accelerates the convergence of the Euler method from first order to second order, akin to RE.
- (∼) The computational cost of BBPN was intermediate between the filtering approach of Schober et al. [2019] and the sampling approaches of Chkrebtii et al. [2016] and Teymur et al. [2016].

In this experiment:

(✓) BBPN is observed to be calibrated.

(✓) BBPN (O2) provides the most accurate approximation among all methods that are calibrated.

(✓) BBPN accelerates the convergence of the Euler method from first order to second order, akin to RE.

(∼) The computational cost of BBPN was intermediate between the filtering approach of Schober et al. [2019] and the sampling approaches of Chkrebtii et al. [2016] and Teymur et al. [2016].

In this experiment:

- (✓) BBPN is observed to be calibrated.
- (✓) BBPN (O2) provides the most accurate approximation among all methods that are calibrated.
- (✓) BBPN accelerates the convergence of the Euler method from first order to second order, akin to RE.
- (∼) The computational cost of BBPN was intermediate between the filtering approach of Schober et al. [2019] and the sampling approaches of Chkrebtii et al. [2016] and Teymur et al. [2016].

Eigenvalue Problems

The calculation of eigenvalues is an important numerical task that had yet to receive attention in PN.

Consider the QR algorithm applued to the following family of 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 BBPN we took:

- $h = 1/\kappa$, where κ is the number of iterations performed.
- the order α is *unknown* and we append it to θ as an additional parameter to be estimated using maximum likelihood.
- eigenvalues are modelled as *a priori* independent (but this might be naïve).

Eigenvalue Problems

The calculation of eigenvalues is an important numerical task that had yet to receive attention in PN.

Consider the QR algorithm applued to the following family of 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 BBPN we took:

- $h = 1/\kappa$, where κ is the number of iterations performed.
- the order α is *unknown* and we append it to θ as an additional parameter to be estimated using maximum likelihood.
- eigenvalues are modelled as *a priori* independent (but this might be naïve).

Eigenvalue Problems

The calculation of eigenvalues is an important numerical task that had yet to receive attention in PN.

Consider the QR algorithm applued to the following family of 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 BBPN we took:

- $h = 1/\kappa$, where κ is the number of iterations performed.
- the order α is *unknown* and we append it to θ as an additional parameter to be estimated using maximum likelihood.
- eigenvalues are modelled as *a priori* independent (but this might be naïve).

Eigenvalue Problems

The calculation of eigenvalues is an important numerical task that had yet to receive attention in PN.

Consider the QR algorithm applued to the following family of 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 BBPN we took:

- $h = 1/\kappa$, where κ is the number of iterations performed.
- the order α is *unknown* and we append it to θ as an additional parameter to be estimated using maximum likelihood.
- eigenvalues are modelled as *a priori* independent (but this might be naïve).

Eigenvalue Problems

The calculation of eigenvalues is an important numerical task that had yet to receive attention in PN.

Consider the QR algorithm applued to the following family of 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 BBPN we took:

- $h = 1/\kappa$, where κ is the number of iterations performed.
- the order α is *unknown* and we append it to θ as an additional parameter to be estimated using maximum likelihood.
- eigenvalues are modelled as *a priori* independent (but this might be naïve).

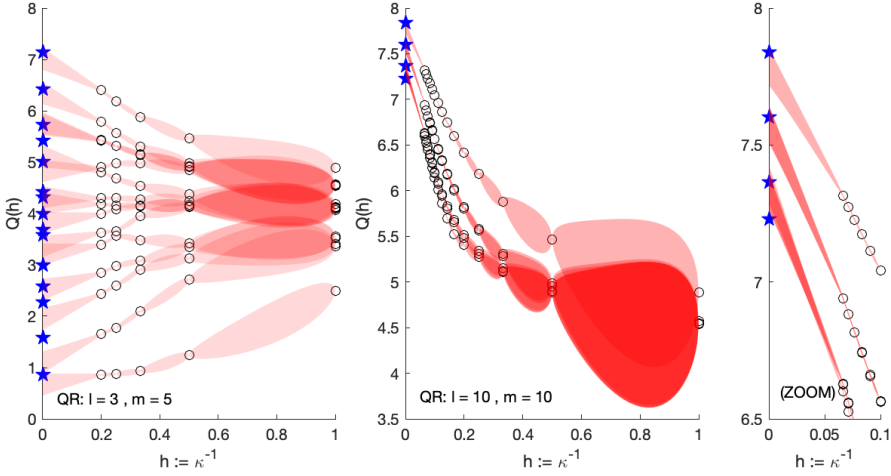

Figure: QR algorithm. All plots show red shaded $\pm 2\sigma$ credible intervals, numerical data as black circles, and true eigenvalues as blue stars. A total of $\kappa = 5$ (left) and 15 (centre) iterations were used.

(✓) No additional computational cost to BBPN, since the dataset is generated during a single run of an iterative numerical method.

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

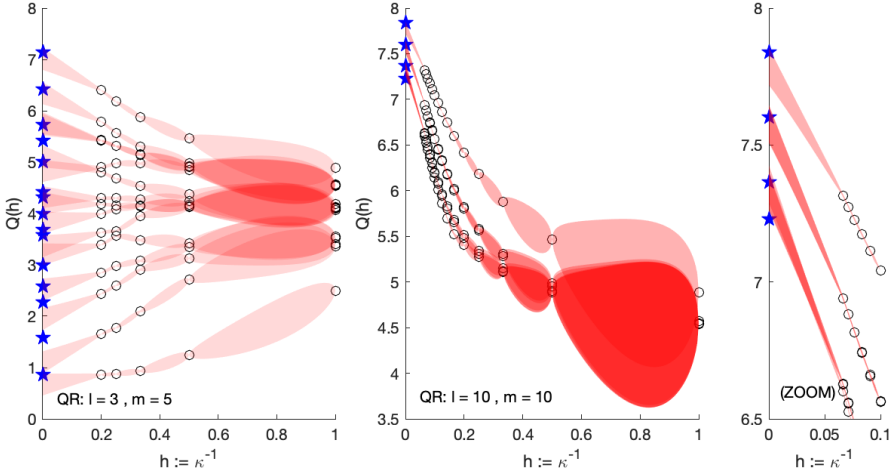

Figure: QR algorithm. All plots show red shaded $\pm 2\sigma$ credible intervals, numerical data as black circles, and true eigenvalues as blue stars. A total of $\kappa = 5$ (left) and 15 (centre) iterations were used.

(✓) No additional computational cost to BBPN, since the dataset is generated during a single run of an iterative numerical method.

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

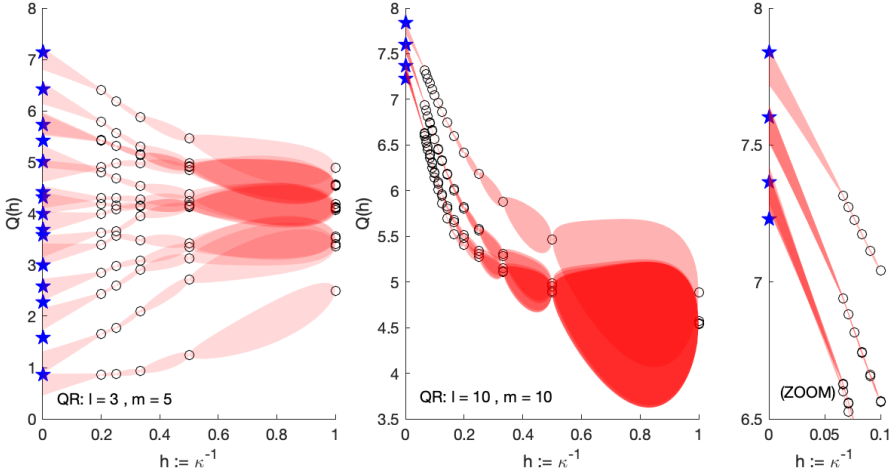

Figure: QR algorithm. All plots show red shaded $\pm 2\sigma$ credible intervals, numerical data as black circles, and true eigenvalues as blue stars. A total of $\kappa = 5$ (left) and 15 (centre) iterations were used.

- (✓) No additional computational cost to BBPN, since the dataset is generated during a single run of an iterative numerical method.
- (✓) Overhead due to fitting the GP is negligible in this experiment.

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

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

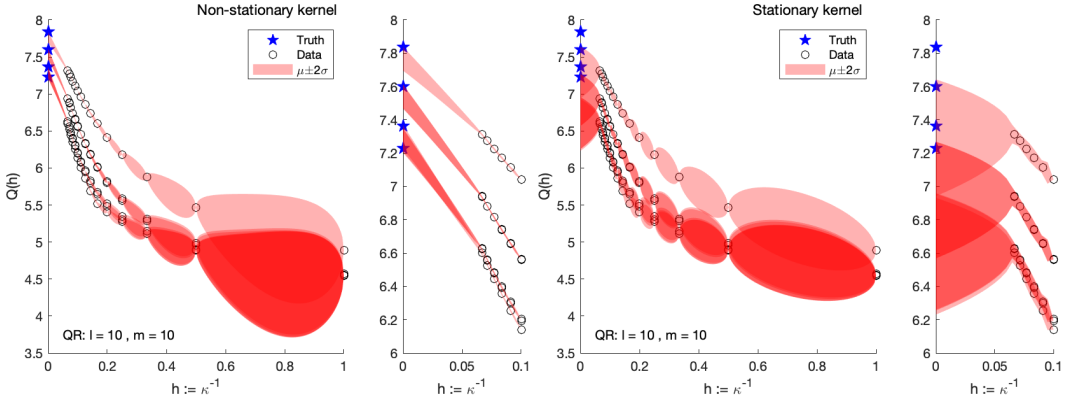

Figure: Comparison of stationary (left) and non-stationary (right) covariance functions for the GP used to model output from the QR algorithm.

Consider the chaotic *Kuramoto–Sivashinsky equation* [Kuramoto, 1978, Sivashinsky, 1977]

$$\partial_t u + \partial_x^4 u + \partial_x^2 u + u\partial_x u = 0,$$

with initial condition $u(x, 0) = \exp(-0.01x^2)$ and periodic boundary conditions on the domain $0 \leq x \leq 1$.

- (apologies - the notation t has been re-purposed!)
- aim to compute $q^*(x) = u(x, 200)$ over the domain $x \in [0, 1]$.
- BBPN was applied to three sequences of five runs of the popular fourth-order time-differencing ETD RK4 numerical scheme [Kassam and Trefethen, 2005], with minimum temporal step size $h = \delta t$ and, for simplicity, a fixed spatial step size $\delta x = 0.001$ throughout.

Consider the chaotic *Kuramoto–Sivashinsky equation* [Kuramoto, 1978, Sivashinsky, 1977]

$$\partial_t u + \partial_x^4 u + \partial_x^2 u + u\partial_x u = 0,$$

with initial condition $u(x, 0) = \exp(-0.01x^2)$ and periodic boundary conditions on the domain $0 \leq x \leq 1$.

- (apologies - the notation t has been re-purposed!)
- aim to compute $q^*(x) = u(x, 200)$ over the domain $x \in [0, 1]$.
- BBPN was applied to three sequences of five runs of the popular fourth-order time-differencing ETD RK4 numerical scheme [Kassam and Trefethen, 2005], with minimum temporal step size $h = \delta t$ and, for simplicity, a fixed spatial step size $\delta x = 0.001$ throughout.

Consider the chaotic *Kuramoto–Sivashinsky equation* [Kuramoto, 1978, Sivashinsky, 1977]

$$\partial_t u + \partial_x^4 u + \partial_x^2 u + u\partial_x u = 0,$$

with initial condition $u(x, 0) = \exp(-0.01x^2)$ and periodic boundary conditions on the domain $0 \leq x \leq 1$.

- (apologies - the notation t has been re-purposed!)
- aim to compute $q^*(x) = u(x, 200)$ over the domain $x \in [0, 1]$.
- BBPN was applied to three sequences of five runs of the popular fourth-order time-differencing ETD RK4 numerical scheme [Kassam and Trefethen, 2005], with minimum temporal step size $h = \delta t$ and, for simplicity, a fixed spatial step size $\delta x = 0.001$ throughout.

Consider the chaotic *Kuramoto–Sivashinsky equation* [Kuramoto, 1978, Sivashinsky, 1977]

$$\partial_t u + \partial_x^4 u + \partial_x^2 u + u\partial_x u = 0,$$

with initial condition $u(x, 0) = \exp(-0.01x^2)$ and periodic boundary conditions on the domain $0 \leq x \leq 1$.

- (apologies - the notation t has been re-purposed!)
- aim to compute $q^*(x) = u(x, 200)$ over the domain $x \in [0, 1]$.
- BBPN was applied to three sequences of five runs of the popular fourth-order time-differencing ETD RK4 numerical scheme [Kassam and Trefethen, 2005], with minimum temporal step size $h = \delta t$ and, for simplicity, a fixed spatial step size $\delta x = 0.001$ throughout.

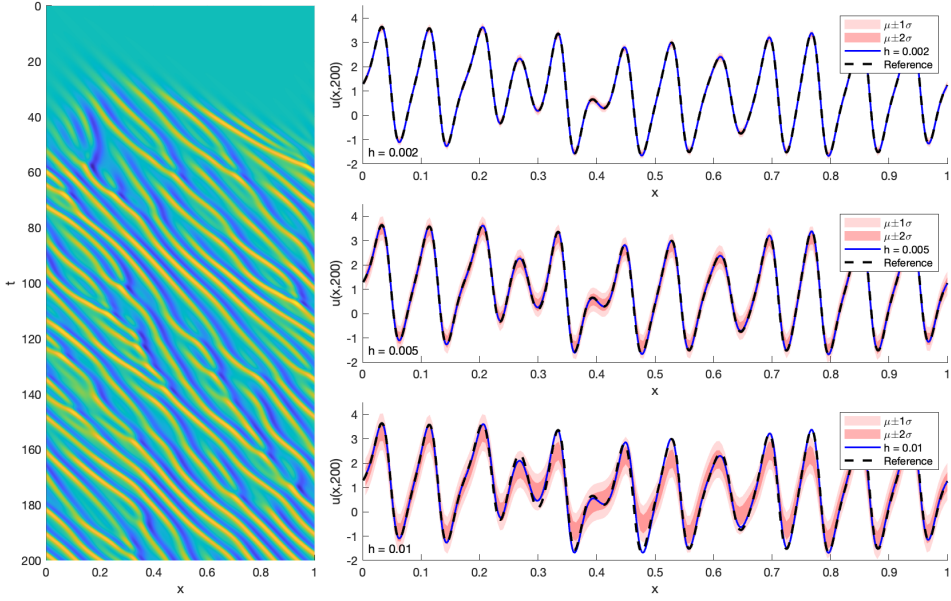

Figure: Partial differential equations. Left: Solution to the Kuramoto–Sivashinsky equation. Right: Approximation of the solution at the final time point ($t = 200$) using BBPN, based on time step sizes $h \in \{0.002, 0.005, 0.01\}$. Posterior mean (blue) and credible regions (shaded) are displayed. A reference solution (dashed black) is obtained by taking $h = 0.0005$.

Conclusion

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

Black Box Probabilistic Numerics, in NeurIPS 2021

Onur Teymur, U. Kent
Chris Foley, Optima Partners
Philip Breen, T. Rowe Price
Toni Karvonen, U. Helsinki

Conclusion

This talk presented *black box probabilistic numerics*, a simple yet powerful framework that bridges the gap between existing PN methods and the numerical state-of-the-art.

The main drawbacks, compared to existing PN:

(✗) a possibly increased computational cost.

(✗) the additional requirement to model the error of a traditional numerical method.

Avenues for further research:

- the use of more flexible and/or computationally cheaper alternatives to GPs.
- experimental design to sequentially select resolutions h_i given an overall computational budget.

`Black Box Probabilistic Numerics, in NeurIPS 2021`

Onur Teymur
U. Kent

Chris Foley
Optima Partners

Philip Breen
T. Rowe Price

Toni Karvonen
U. Helsinki

References I

S. Bartels and P. Hennig. Probabilistic approximate least-squares. *Proc. 19th Int. Conf. Artificial Intelligence and Statistics*, 51:676–684, 2016.

S. Bartels, J. Cockayne, I. C. Ipsen, and P. Hennig. Probabilistic linear solvers: a unifying view. *Stat. Comput.*, 29(6):1249–1263, 2019.

N. Bosch, P. Hennig, and F. Tronarp. Calibrated Adaptive Probabilistic ODE Solvers. *PMLR*, 130:3466–3474, 2021. ISSN 2640-3498.

F.-X. Briol, C. J. Oates, M. Girolami, and M. A. Osborne. Frank-Wolfe Bayesian quadrature: probabilistic integration with theoretical guarantees. *NeurIPS 28*, 2015.

F.-X. Briol, C. J. Oates, M. Girolami, M. A. Osborne, D. Sejdinovic, et al. Probabilistic integration: a role in statistical computation? *Stat. Sci.*, 34(1):1–22, 2019.

R. Bulirsch and J. Stoer. Fehlerabschätzungen und Extrapolation mit rationalen Funktionen bei Verfahren vom Richardson-Typus. *Numer. Math.*, 6(1):413–427, 1964.

H. R. Chai and R. Garnett. Improving quadrature for constrained integrands. *PMLR*, 89:2751–2759, 2019.

O. A. Chkrebtii and D. A. Campbell. Adaptive step-size selection for state-space probabilistic differential equation solvers. *Stat. Comput.*, 29(6):1285–1295, 2019. ISSN 1573-1375.

O. A. Chkrebtii, D. A. Campbell, B. Calderhead, and M. A. Girolami. Bayesian solution uncertainty quantification for differential equations. *Bayesian Anal.*, 11(4):1239–1267, 2016.

J. Cockayne, C. Oates, T. Sullivan, and M. Girolami. Probabilistic numerical methods for PDE-constrained Bayesian inverse problems. *AIP Conference Proceedings*, 1853:060001, 2017.

J. Cockayne, C. J. Oates, I. C. Ipsen, M. Girolami, et al. A Bayesian conjugate gradient method (with discussion). *Bayesian Anal.*, 14(3):937–1012, 2019.

J. Cockayne, I. C. Ipsen, C. J. Oates, and T. W. Reid. Probabilistic iterative methods for linear systems. *Journal of Machine Learning Research*, 22(232):1–34, 2021.

References II

P. R. Conrad, M. Girolami, S. Särkkä, A. Stuart, and K. Zygalakis. Statistical analysis of differential equations: Introducing probability measures on numerical solutions. *Stat. Comput.*, 27(4):1065–1082, 2016.

P. Diaconis. Bayesian numerical analysis. *Statistical Decision Theory and Related Topics IV*, 1:163–175, 1988.

M. Fisher, C. Oates, C. Powell, and A. Teckentrup. A locally adaptive Bayesian cubature method. *PMLR*, 108: 1265–1275, 2020.

A. Gessner, J. Gonzalez, and M. Mahsereci. Active multi-information source Bayesian quadrature. *PMLR*, 115: 712–721, 2020.

T. Gunter, M. A. Osborne, R. Garnett, P. Hennig, and S. J. Roberts. Sampling for inference in probabilistic models with fast Bayesian quadrature. *NeurIPS 27*, 2014.

P. Hennig. Probabilistic interpretation of linear solvers. *SIAM J. Optim.*, 25(1):234–260, 2015.

P. Hennig and S. Hauberg. Probabilistic solutions to differential equations and their application to Riemannian statistics. *PMLR*, 33:347–355, 2013.

P. Hennig and M. Kiefel. Quasi-Newton methods: A new direction. *JMLR*, 14(1):843–865, 2013. ISSN 1532-4435.

R. Jagadeeswaran and F. J. Hickernell. Fast automatic Bayesian cubature using lattice sampling. *Stat. Comput.*, 29(6):1215–1229, 2019.

D. C. Joyce. Survey of extrapolation processes in numerical analysis. *SIAM Rev.*, 13(4):435–490, 1971.

T. Karvonen and C. J. Oates. Maximum likelihood estimation in Gaussian process regression is ill-posed. *arXiv:2203.09179*, 2022.

T. Karvonen and S. Särkkä. Classical quadrature rules via Gaussian processes. *IEEE 27th Int. Workshop on Machine Learning for Signal Processing*, 2017.

T. Karvonen, C. J. Oates, and S. Särkkä. A Bayes–Sard cubature method. *NeurIPS 31*, 2018.

References III

T. Karvonen, S. Särkkä, and C. J. Oates. Symmetry exploits for Bayesian cubature methods. *Stat. Comput.*, 29 (6):1231–1248, 2019.

A. Kassam and L. N. Trefethen. Fourth-order time stepping for stiff PDEs. *SIAM J. Sci. Comput.*, 26: 1214–1233, 2005.

M. Kennedy. Bayesian quadrature with non-normal approximating functions. *Stat. Comput.*, 8(4):365–375, 1998.

H. Kersting and P. Hennig. Active uncertainty calibration in Bayesian ODE solvers. *Proc. 32nd Conf. Uncertainty in Artificial Intelligence*, pages 309–318, 2016.

H. Kersting, T. J. Sullivan, and P. Hennig. Convergence rates of Gaussian ODE filters. *Stat. Comput.*, 30(6): 1791–1816, 2020.

Y. Kuramoto. Diffusion-induced chaos in reaction systems. *Prog. Theor. Phys. Supp.*, 64:346–367, 1978.

F. Larkin. Gaussian measure in Hilbert space and applications in numerical analysis. *Rocky Mountain J. Math.*, 2(3):379–422, 1972.

M. Mahsereci and P. Hennig. Probabilistic line searches for stochastic optimization. *NeurIPS 28*, 2015.

J. Mockus. On Bayesian methods for seeking the extremum and their application. *IFIP Congress*, pages 195–200, 1977.

J. Mockus. *Bayesian Approach to Global Optimization*. Springer, 1989.

J. Mockus, V. Tiesis, and A. Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2(2):117–129, 1978.

A. O'Hagan. Bayes–Hermite quadrature. *J. Stat. Plan. Infer.*, 29(3):245–260, 1991.

A. O'Hagan. Some Bayesian numerical analysis. *Bayesian Statistics 4: Proc. 4th Valencia International Meeting*, pages 345–363, 1992.

References IV

M. Osborne, R. Garnett, S. Roberts, C. Hart, S. Aigrain, and N. Gibson. Bayesian quadrature for ratios. *PMLR*, 22:832–840, 2012.

H. Owhadi. Bayesian numerical homogenization. *Multiscale Model. Sim.*, 13(3):812–828, 2015.

H. Owhadi and C. Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization: From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*. Cambridge University Press, 2019.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007.

J. Prüher and S. Särkkä. On the use of gradient information in Gaussian process quadratures. In *IEEE 26th Int. Workshop on Machine Learning for Signal Processing*, 2016.

C. E. Rasmussen and Z. Ghahramani. Bayesian Monte Carlo. *NeurIPS 15*, pages 505–512, 2003.

T. W. Reid, I. C. Ipsen, J. Cockayne, and C. J. Oates. A probabilistic numerical extension of the conjugate gradient method. *arXiv:2008.03225*, 2020.

C. Runge. Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten. *Zeitschrift für Mathematik und Physik*, 46(224–243):20, 1901.

F. Schäfer, T. J. Sullivan, and H. Owhadi. Compression, inversion, and approximate PCA of dense kernel matrices at near-linear computational complexity. *Multiscale Model. Simul.*, 19(2):688–730, 2021.

M. Schober, D. Duvenaud, and P. Hennig. Probabilistic ODE solvers with Runge-Kutta means. *NeurIPS 27*, 2014.

M. Schober, S. Särkkä, and P. Hennig. A probabilistic model for the numerical solution of initial value problems. *Stat. Comput.*, 29(1):99–122, 2019.

G. Sivashinsky. Nonlinear analysis of hydrodynamic instability in laminar flames. *Acta Astronaut.*, 4(11): 1177–1206, 1977.

References V

J. Skilling. Bayesian solution of ordinary differential equations. *Maximum Entropy and Bayesian Methods*, pages 23–37, 1992. doi: 10.1007/978-94-017-2219-3_2.

J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. *NeurIPS 25*, 2012.

M. L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, 2012.

O. Teymur, K. Zygalakis, and B. Calderhead. Probabilistic linear multistep methods. *NeurIPS 29*, 2016.

O. Teymur, B. Calderhead, H. C. Lie, and T. J. Sullivan. Implicit probabilistic integrators for ODEs. *NeurIPS 31*, 2018.

F. Tronarp, H. Kersting, S. Särkkä, and P. Hennig. Probabilistic solutions to ordinary differential equations as nonlinear Bayesian filtering: a new perspective. *Stat. Comput.*, 29(6):1297–1315, 2019.

J. Wang, J. Cockayne, and C. Oates. On the Bayesian solution of differential equations. *arXiv:1805.07109*, 2018.

J. Wang, J. Cockayne, O. Chkrebtii, T. J. Sullivan, and C. J. Oates. Bayesian numerical methods for nonlinear partial differential equations. *arXiv:2104.12587*, 2021.

J. Wenger and P. Hennig. Probabilistic linear solvers for machine learning. *NeurIPS 34*, 2020.

X. Xi, F.-X. Briol, and M. Girolami. Bayesian quadrature for multiple related integrals. *PMLR*, 80:5373–5382, 2018.