# Introducing time parallelisation within data assimilation using parareal

Rishabh Bhatt, Laurent Debreu, Arthur Vidard

Université Grenoble Alpes, INRIA, Laboratoire Jean Kuntzmann

11th Parallel-in-Time Conference

CIRM, July 15, 2022


**Motivation**

- Numerical weather prediction (NWP) heavily relies on the use of high performance architectures to give forecasts in reasonable time
- Example: A climate simulation problem with $50,000$ unknowns given $10,000$ cores. Using $10 \times 10$ unknowns for the horizontal domain would use 500 cores.
  - More cores $\implies$ performance loss
- Data assimilation (DA) is a highly sequential algorithm, why not exploit PinT methods...?

**In this talk**

- Coupling parareal with the data assimilation framework
- Tuning the precision levels in minimisation for running parareal adaptively
- results with a 1D shallow-water model and a theoretical speedup check
- Some of the possible improvements and future work

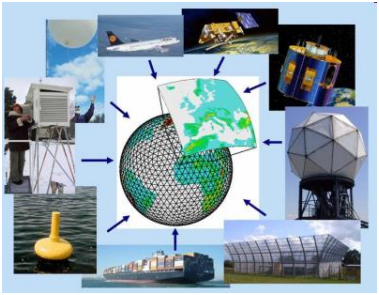
**Motivation**

- Numerical weather prediction (NWP) heavily relies on the use of high performance architectures to give forecasts in reasonable time
- Example: A climate simulation problem with $50,000$ unknowns given $10,000$ cores. Using $10 \times 10$ unknowns for the horizontal domain would use 500 cores.
  - More cores $\implies$ performance loss
- Data assimilation (DA) is a highly sequential algorithm, why not exploit PinT methods...?

**In this talk**

- Coupling parareal with the data assimilation framework
- Tuning the precision levels in minimisation for running parareal adaptively
- results with a 1D shallow-water model and a theoretical speedup check
- Some of the possible improvements and future work

In terms of NWP, the aim is to retrieve the *optimal* initial condition which describes the true state of the atmosphere to give an improved forecast

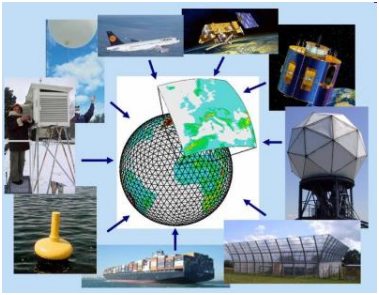


source: dwd.de

In terms of NWP, the aim is to retrieve the *optimal* initial condition which describes the true state of the atmosphere to give an improved forecast

Available information (pressure, temperature, wind speed,...)

- numerical model describing the governing laws/physics
  - state vector of order $10^8 - 10^9$
- in-situ measurements, satellite data etc
  - observation vector of order $10^6 - 10^7$



source: dwd.de

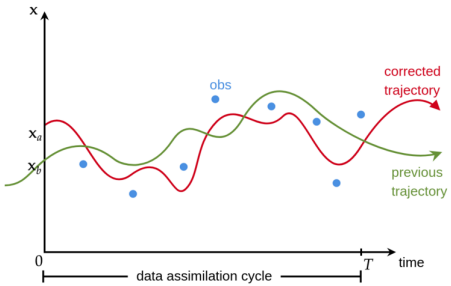
Discrete nonlinear model $\mathcal{F}$ with state variable $\mathbf{x}$

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_i &= \mathcal{F}_{t_{i-1} \to t_i}(\mathbf{x}_{i-1}) \quad i = 1, ..., N \qquad N \text{ - no. of time windows} \end{aligned} \tag{1}$$

Discrete nonlinear model $\mathcal{F}$ with state variable $\mathbf{x}$

$$\mathbf{x}(t_0) = \mathbf{x}_0$$
$$\mathbf{x}_i = \mathcal{F}_{t_{i-1} \to t_i}(\mathbf{x}_{i-1}) \quad i = 1, ..., N \qquad N \text{ - no. of time windows} \tag{1}$$

**Variational DA** $\longrightarrow$ solving a least squares problem of the discrepancies between what model gives and what we observe!

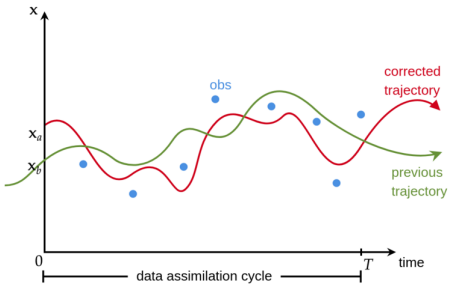


Discrete nonlinear model $\mathcal{F}$ with state variable $\mathbf{x}$

$$\mathbf{x}(t_0) = \mathbf{x}_0$$
$$\mathbf{x}_i = \mathcal{F}_{t_{i-1} \to t_i}(\mathbf{x}_{i-1}) \quad i = 1, ..., N \qquad N \text{ - no. of time windows} \tag{1}$$

**Variational DA** $\longrightarrow$ solving a least squares problem of the discrepancies between what model gives and what we observe!

- background state $\mathbf{x}_0^b$
- observations $\mathbf{y}_i$
- observation operator $\mathcal{H}$
- covariance matrices $\mathbf{B}$ and $\mathbf{R}$



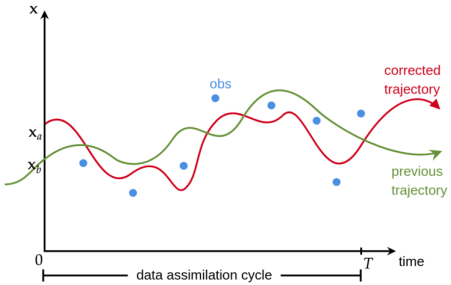

$$\min_{\mathbf{x}_0 \in \mathbb{R}^n} \mathcal{J}(\mathbf{x}_0) = \underbrace{\frac{1}{2}\|\mathbf{x}_0 - \mathbf{x}_0^b\|^2_{\mathbf{B}^{-1}}}_{\mathcal{J}_b(\mathbf{x}_0)} + \underbrace{\frac{1}{2}\sum_{i=0}^{N} \|\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)\|^2_{\mathbf{R}_i^{-1}}}_{\mathcal{J}_o(\mathbf{x}_0)} \tag{2}$$

Discrete nonlinear model $\mathcal{F}$ with state variable $\mathbf{x}$

$$\mathbf{x}(t_0) = \mathbf{x}_0$$
$$\mathbf{x}_i = \mathcal{F}_{t_{i-1} \to t_i}(\mathbf{x}_{i-1}) \quad i = 1, ..., N \qquad N \text{ - no. of time windows} \tag{1}$$

**Variational DA** $\longrightarrow$ solving a least squares problem of the discrepancies between what model gives and what we observe!

- background state $\mathbf{x}_0^b$
- observations $\mathbf{y}_i$
- observation operator $\mathcal{H}$
- covariance matrices $\mathbf{B}$ and $\mathbf{R}$




$$\min_{\mathbf{x}_0 \in \mathbb{R}^n} \mathcal{J}(\mathbf{x}_0) = \underbrace{\frac{1}{2}\|\mathbf{x}_0 - \mathbf{x}_0^b\|^2_{\mathbf{B}^{-1}}}_{\mathcal{J}_b(\mathbf{x}_0)} + \underbrace{\frac{1}{2}\sum_{i=0}^{N} \|\mathbf{y}_i - \mathcal{H}_i(\mathcal{F}_{t_0 \to t_i}(\mathbf{x}_0))\|^2_{\mathbf{R}_i^{-1}}}_{\mathcal{J}_o(\mathbf{x}_0)} \tag{2}$$

minimisation algorithms involve lots of function and gradient evaluations

$$\mathcal{J}(\mathbf{x}_0) \rightarrow \text{one model run}$$

$$\nabla\mathcal{J}(\mathbf{x}_0) = \begin{pmatrix} \dfrac{\partial\mathcal{J}}{\partial x_1}(\mathbf{x}_0) \\ \vdots \\ \dfrac{\partial\mathcal{J}}{\partial x_n}(\mathbf{x}_0) \end{pmatrix} \simeq \begin{pmatrix} \dfrac{\mathcal{J}(\mathbf{x}_0+\alpha\mathbf{e}_1)-\mathcal{J}(\mathbf{x}_0)}{\alpha} \\ \vdots \\ \dfrac{\mathcal{J}(\mathbf{x}_0+\alpha\mathbf{e}_n)-\mathcal{J}(\mathbf{x}_0)}{\alpha} \end{pmatrix} \rightarrow n+1 \,\text{model runs}$$

**Tangent linear model** of $\mathcal{F}$ (for initial perturbation $\delta\mathbf{x}_0$)

$$\delta\mathbf{x}_{i+1} = F_i\delta\mathbf{x}_i$$

**Corresponding adjoint equations:**

$$\begin{aligned} \lambda_N &= H_N^T\mathbf{R}_N^{-1}[\mathbf{y}_N - \mathcal{H}_N(\mathbf{x}_N)] \\ \lambda_i &= F_i^T\lambda_{i+1} + H_i^T\mathbf{R}_i^{-1}[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)] \qquad i = N-1,\ldots,1 \\ \lambda_0 &= F_0^T\lambda_1 + \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + H_0^T\mathbf{R}_0^{-1}[\mathbf{y}_0 - \mathcal{H}_0(\mathbf{x}_0)] \end{aligned} \tag{3}$$

Solving (3) gives $\delta\mathcal{J} = (\lambda_0)^T\delta\mathbf{x}_0$ $\rightsquigarrow$ cost around 3 times running forward model

minimisation algorithms involve lots of function and gradient evaluations

$$\mathcal{J}(\mathbf{x}_0) \rightarrow \text{one model run}$$

$$\nabla\mathcal{J}(\mathbf{x}_0) = \begin{pmatrix} \dfrac{\partial\mathcal{J}}{\partial x_1}(\mathbf{x}_0) \\ \vdots \\ \dfrac{\partial\mathcal{J}}{\partial x_n}(\mathbf{x}_0) \end{pmatrix} \simeq \begin{pmatrix} \dfrac{\mathcal{J}(\mathbf{x}_0+\alpha\mathbf{e}_1)-\mathcal{J}(\mathbf{x}_0)}{\alpha} \\ \vdots \\ \dfrac{\mathcal{J}(\mathbf{x}_0+\alpha\mathbf{e}_n)-\mathcal{J}(\mathbf{x}_0)}{\alpha} \end{pmatrix} \rightarrow n+1 \,\text{model runs}$$

**Tangent linear model** of $\mathcal{F}$ (for initial perturbation $\delta\mathbf{x}_0$)

$$\delta\mathbf{x}_{i+1} = F_i\delta\mathbf{x}_i$$

**Corresponding adjoint equations:**

$$\begin{aligned} \lambda_N &= H_N^T\mathbf{R}_N^{-1}[\mathbf{y}_N - \mathcal{H}_N(\mathbf{x}_N)] \\ \lambda_i &= F_i^T\lambda_{i+1} + H_i^T\mathbf{R}_i^{-1}[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)] \qquad i = N-1,\ldots,1 \\ \lambda_0 &= F_0^T\lambda_1 + \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + H_0^T\mathbf{R}_0^{-1}[\mathbf{y}_0 - \mathcal{H}_0(\mathbf{x}_0)] \end{aligned} \tag{3}$$

Solving (3) gives $\delta\mathcal{J} = (\lambda_0)^T\delta\mathrm{x}_0$ $\rightsquigarrow$ cost around 3 times running forward model

minimisation algorithms involve lots of function and gradient evaluations

$$\mathcal{J}(\mathbf{x}_0) \rightarrow \text{one model run}$$

$$\nabla\mathcal{J}(\mathbf{x}_0) = \begin{pmatrix} \dfrac{\partial\mathcal{J}}{\partial x_1}(\mathbf{x}_0) \\ \vdots \\ \dfrac{\partial\mathcal{J}}{\partial x_n}(\mathbf{x}_0) \end{pmatrix} \simeq \begin{pmatrix} \dfrac{\mathcal{J}(\mathbf{x}_0 + \alpha\mathbf{e}_1) - \mathcal{J}(\mathbf{x}_0)}{\alpha} \\ \vdots \\ \dfrac{\mathcal{J}(\mathbf{x}_0 + \alpha\mathbf{e}_n) - \mathcal{J}(\mathbf{x}_0)}{\alpha} \end{pmatrix} \rightarrow n+1 \,\text{model runs}$$

**Tangent linear model** of $\mathcal{F}$ (for initial perturbation $\delta\mathbf{x}_0$)

$$\delta\mathbf{x}_{i+1} = F_i\delta\mathbf{x}_i$$

**Corresponding adjoint equations:**

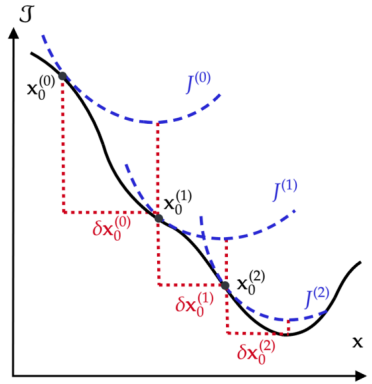
$$\begin{aligned} \lambda_N &= H_N^T\mathbf{R}_N^{-1}[\mathbf{y}_N - \mathcal{H}_N(\mathbf{x}_N)] \\ \lambda_i &= F_i^T\lambda_{i+1} + H_i^T\mathbf{R}_i^{-1}[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)] \qquad i = N-1,\ldots,1 \\ \lambda_0 &= F_0^T\lambda_1 + \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + H_0^T\mathbf{R}_0^{-1}[\mathbf{y}_0 - \mathcal{H}_0(\mathbf{x}_0)] \end{aligned} \tag{3}$$

Solving (3) gives $\delta\mathcal{J} = (\lambda_0)^T\delta\mathbf{x}_0$ $\rightsquigarrow$ cost around 3 times running forward model

- Solve small sub-minimisations by sucessive approximations
- Gives a quadratic $J$

**Tangent linear approximations** (around $\mathbf{x}^{(k)}$)

$$\mathcal{F}(\mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}) \simeq \mathcal{F}(\mathbf{x}^{(k)}) + F|_{\mathbf{x}^{(k)}}\, \delta\mathbf{x}^{(k)}$$
$$\mathcal{H}(\mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}) \simeq \mathcal{H}(\mathbf{x}^{(k)}) + H|_{\mathcal{F}(\mathbf{x}^{(k)})}\delta\mathbf{x}^{(k)} \quad (4)$$




- At $k$th minimisation solve

$$J^k(\delta\mathbf{x}_0^{(k)}) = \frac{1}{2}\|\delta\mathbf{x}_0^{(k)} - (\mathbf{x}_0^b - \mathbf{x}_0^{(k)})\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{i=0}^{N}\|\mathbf{y}_i - \mathcal{H}_i \circ \mathcal{F}(\mathbf{x}_0^{(k)} + \delta\mathbf{x}_0^{(k)})\|^2_{\mathbf{R}^{-1}} \quad (5)$$

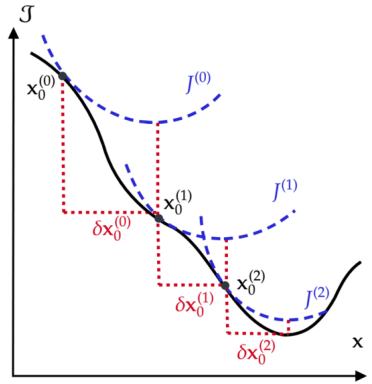
  - Innovation vector : $\mathbf{d}_i = \mathbf{y}_i - \mathcal{H}_i \circ \mathcal{F}(\mathbf{x}_0^{(k)})$
  - Increment vector : $\delta\mathbf{x}_0^{(k)} = \mathbf{x}_0^{(k+1)} - \mathbf{x}_0^{(k)}$

[1] Courtier, Thépaut, Hollingsworth (1994)

- Solve small sub-minimisations by sucessive approximations
- Gives a quadratic $J$

**Tangent linear approximations** (around $\mathbf{x}^{(k)}$)

$$\mathcal{F}(\mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}) \simeq \mathcal{F}(\mathbf{x}^{(k)}) + F|_{\mathbf{x}^{(k)}}\, \delta\mathbf{x}^{(k)}$$
$$\mathcal{H}(\mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}) \simeq \mathcal{H}(\mathbf{x}^{(k)}) + H|_{\mathcal{F}(\mathbf{x}^{(k)})}\delta\mathbf{x}^{(k)} \quad (4)$$




- At $k$th minimisation solve

$$J^k(\delta\mathbf{x}_0^{(k)}) = \frac{1}{2}\|\delta\mathbf{x}_0^{(k)} - (\mathbf{x}_0^b - \mathbf{x}_0^{(k)})\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{i=0}^{N}\|\mathbf{y}_i - \mathcal{H}_i \circ \mathcal{F}(\mathbf{x}_0^{(k)} + \delta\mathbf{x}_0^{(k)})\|^2_{\mathbf{R}^{-1}} \quad (5)$$

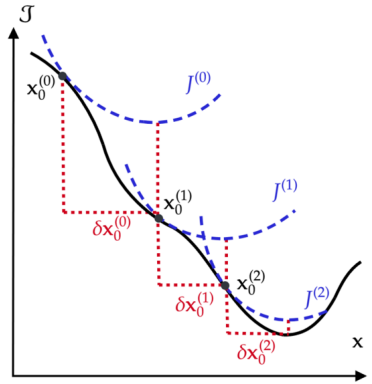
  - Innovation vector : $\mathbf{d}_i = \mathbf{y}_i - \mathcal{H}_i \circ \mathcal{F}(\mathbf{x}_0^{(k)})$
  - Increment vector : $\delta\mathbf{x}_0^{(k)} = \mathbf{x}_0^{(k+1)} - \mathbf{x}_0^{(k)}$

[1] Courtier, Thépaut, Hollingsworth (1994)

- Solve small sub-minimisations by sucessive approximations
- Gives a quadratic $J$

**Tangent linear approximations** (around $\mathbf{x}^{(k)}$)

$$\mathcal{F}(\mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}) \simeq \mathcal{F}(\mathbf{x}^{(k)}) + F|_{\mathbf{x}^{(k)}}\, \delta\mathbf{x}^{(k)}$$
$$\mathcal{H}(\mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}) \simeq \mathcal{H}(\mathbf{x}^{(k)}) + H|_{\mathcal{F}(\mathbf{x}^{(k)})}\delta\mathbf{x}^{(k)} \quad (4)$$




- At $k$th minimisation solve

$$J^k(\delta\mathbf{x}_0^{(k)}) = \frac{1}{2}\|\delta\mathbf{x}_0^{(k)} - (\mathbf{x}_0^b - \mathbf{x}_0^{(k)})\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{i=0}^{N}\|\mathbf{d}_i - H_i F \delta\mathbf{x}_0^{(k)}\|^2_{\mathbf{R}^{-1}} \quad (5)$$

  - Innovation vector : $\mathbf{d}_i = \mathbf{y}_i - \mathcal{H}_i \circ \mathcal{F}(\mathbf{x}_0^{(k)})$
  - Increment vector : $\delta\mathbf{x}_0^{(k)} = \mathbf{x}_0^{(k+1)} - \mathbf{x}_0^{(k)}$

[1] Courtier, Thépaut, Hollingsworth (1994)

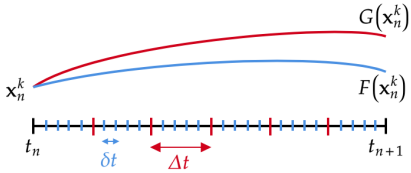
Discrete linear model

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_{i+1} &= F\mathbf{x}_i \quad i = 1, ..., N \end{aligned} \qquad N \text{ - no. of time windows} \tag{6}$$

**Description:**

$$\mathbf{x}_0^{k+1} = \mathbf{x}^0$$

$$\mathbf{x}_{n+1}^{k+1} = \underbrace{G\,\mathbf{x}_n^{k+1}}_{\text{prediction}} + \underbrace{(F-G)\,\mathbf{x}_n^k}_{\text{correction}}$$




- Fine solver computations in parallel
- Nice convergence for parabolic problems, not so good for hyperbolic

**Error:**

With parareal integrator $P(k)$ at iteration $k$ such that $P(k)\mathbf{x}_0 = \mathbf{x}_N^k$, we can write [2]

$$F^N - P(k) = \sum_{p=k+1}^{N} C_p^N (F-G)^p G^{N-p} \tag{7}$$

[2] Staff and Ronquist

[3] J. Lions, Y. Maday, and G. Turinici (2001)

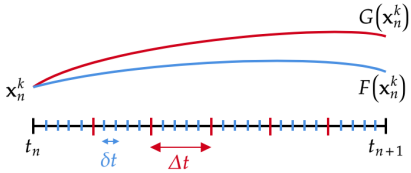
Discrete linear model

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_{i+1} &= F\mathbf{x}_i \quad i = 1, ..., N \end{aligned} \qquad N \text{ - no. of time windows} \tag{6}$$

**Description:**

$$\mathbf{x}_0^{k+1} = \mathbf{x}^0$$

$$\mathbf{x}_{n+1}^{k+1} = \underbrace{G\,\mathbf{x}_n^{k+1}}_{\text{prediction}} + \underbrace{(F-G)\,\mathbf{x}_n^k}_{\text{correction}}$$




- Fine solver computations in parallel
- Nice convergence for parabolic problems, not so good for hyperbolic

**Error:**

With parareal integrator $P(k)$ at iteration $k$ such that $P(k)\mathbf{x}_0 = \mathbf{x}_N^k$, we can write [2]

$$F^N - P(k) = \sum_{p=k+1}^{N} C_p^N (F-G)^p G^{N-p} \tag{7}$$

[2] Staff and Ronquist

[3] J. Lions, Y. Maday, and G. Turinici (2001)

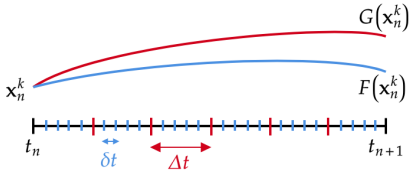
Discrete linear model

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_{i+1} &= F\mathbf{x}_i \quad i = 1, ..., N \end{aligned} \qquad N \text{ - no. of time windows} \tag{6}$$

**Description:**

$$\mathbf{x}_0^{k+1} = \mathbf{x}^0$$

$$\mathbf{x}_{n+1}^{k+1} = \underbrace{G\,\mathbf{x}_n^{k+1}}_{\text{prediction}} + \underbrace{(F-G)\,\mathbf{x}_n^k}_{\text{correction}}$$




- Fine solver computations in parallel
- Nice convergence for parabolic problems, not so good for hyperbolic

**Error:**

With parareal integrator $P(k)$ at iteration $k$ such that $P(k)\mathbf{x}_0 = \mathbf{x}_N^k$, we can write [2]

$$F^N - P(k) = \sum_{p=k+1}^{N} C_p^N (F-G)^p G^{N-p} \tag{7}$$

[2] Staff and Ronquist

[3] J. Lions, Y. Maday, and G. Turinici (2001)

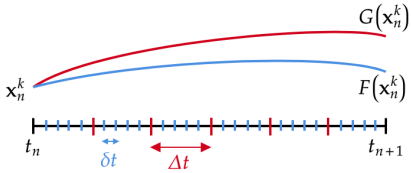
Discrete linear model

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_{i+1} &= F\mathbf{x}_i \quad i = 1, ..., N \end{aligned} \qquad N \text{ - no. of time windows} \tag{6}$$

**Description:**

$$\mathbf{x}_0^{k+1} = \mathbf{x}^0$$

$$\mathbf{x}_{n+1}^{k+1} = \underbrace{G\,\mathbf{x}_n^{k+1}}_{\text{prediction}} + \underbrace{(F-G)\,\mathbf{x}_n^k}_{\text{correction}}$$




- Fine solver computations in parallel
- Nice convergence for parabolic problems, not so good for hyperbolic

**Error:**

With parareal integrator $P(k)$ at iteration $k$ such that $P(k)\mathbf{x}_0 = \mathbf{x}_N^k$, we can write [2]

$$F^N - P(k) = \sum_{p=k+1}^{N} C_p^N (F-G)^p G^{N-p} \tag{7}$$

[2] Staff and Ronquist

[3] J. Lions, Y. Maday, and G. Turinici (2001)

**Data assimilation problem**: with no background info ($\mathbf{x}_0^b$), a true observation $\mathbf{y}$ at $t_N = T$ and $H = I$.

**Data assimilation problem**: with no background info ($\mathbf{x}_0^b$), a true observation $\mathbf{y}$ at $t_N = T$ and $H = I$.

$$J(\mathbf{x}_0) = \frac{1}{2}\|F^N\mathbf{x}_0 - \mathbf{y}\|_2^2$$

$$\nabla J(\mathbf{x}_0) = (F^N)^T(F^N\mathbf{x}_0 - \mathbf{y})$$

**Data assimilation problem**: with no background info ($\mathbf{x}_0^b$), a true observation $\mathbf{y}$ at $t_N = T$ and $H = I$.

$$J(\mathbf{x}_0) = \frac{1}{2}\|F^N\mathbf{x}_0 - \mathbf{y}\|_2^2$$

$$\nabla J(\mathbf{x}_0) = (F^N)^T(F^N\mathbf{x}_0 - \mathbf{y})$$

Optimality condition: $\nabla J(\mathbf{x}_0) = 0 \implies$ Solving linear system $\underbrace{(F^N)^T F^N}_{A^*}\mathbf{x}_0 = \underbrace{(F^N)^T\mathbf{y}}_{\mathbf{b}}$

**Data assimilation problem**: with no background info ($\mathbf{x}_0^b$), a true observation $\mathbf{y}$ at $t_N = T$ and $H = I$.

$$J(\mathbf{x}_0) = \frac{1}{2}\|F^N\mathbf{x}_0 - \mathbf{y}\|_2^2$$

$$\nabla J(\mathbf{x}_0) = (F^N)^T(F^N\mathbf{x}_0 - \mathbf{y})$$

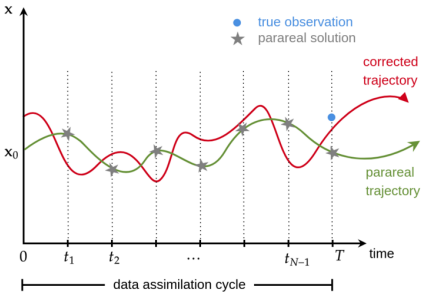
Optimality condition: $\nabla J(\mathbf{x}_0) = 0 \implies$ Solving linear system $\underbrace{(F^N)^T F^N}_{A^*}\mathbf{x}_0 = \underbrace{(F^N)^T\mathbf{y}}_{\mathbf{b}}$

**Idea**:

Parareal for forward integration i.e. $F^N \approx P(k)$

$$\nabla J(\mathbf{x}_0) \approx (F^N)^T(P(k)\,\mathbf{x}_0 - \mathbf{y}) \qquad (8)$$

$$\text{Solve} \longrightarrow \underbrace{(F^N)^T P(k)}_{A}\mathbf{x}_0 = \underbrace{(F^N)^T\mathbf{y}}_{\mathbf{b}}$$



**Data assimilation problem**: with no background info ($\mathbf{x}_0^b$), a true observation $\mathbf{y}$ at $t_N = T$ and $H = I$.

$$J(\mathbf{x}_0) = \frac{1}{2}\|F^N\mathbf{x}_0 - \mathbf{y}\|_2^2$$

$$\nabla J(\mathbf{x}_0) = (F^N)^T(F^N\mathbf{x}_0 - \mathbf{y})$$

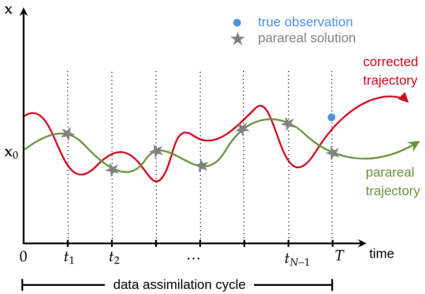
Optimality condition: $\nabla J(\mathbf{x}_0) = 0 \implies$ Solving linear system $\underbrace{(F^N)^T F^N}_{A^*}\mathbf{x}_0 = \underbrace{(F^N)^T\mathbf{y}}_{\mathbf{b}}$

**Idea**:

Parareal for forward integration i.e. $F^N \approx P(k)$

$$\nabla J(\mathbf{x}_0) \approx (F^N)^T(P(k)\,\mathbf{x}_0 - \mathbf{y}) \qquad (8)$$

$$\text{Solve} \longrightarrow \underbrace{(F^N)^T P(k)}_{A}\mathbf{x}_0 = \underbrace{(F^N)^T\mathbf{y}}_{\mathbf{b}}$$




$\longrightarrow$ Conjugate gradient for minimisation as $A^*$ is symmetric

# Minimisation: Inexact conjugate gradient[4]

- *Inexact* matrix-vector multiplication (a costly operation!)
- By gradually increasing the inexactness of the mat-vec product, one could still produce the exact version's level of accuracy.
- In our case, the perturbation matrix is

$$E(k) = A - A^* = -(F^N)^T(F^N - P(k))$$

**Theorem in nutshell (a bit modified...)**: At icg-iteration $j$ if

$$\|E_j p_j\|_{A^{-1}} \leq \omega_j \left(\|\mathbf{b}\|_{A^{-1}}, \|\mathbf{p}_j\|_A, \|\mathbf{r}\|_{A^{-1}}\right) \tag{9}$$

then

$$\| \underbrace{\mathbf{r}(\mathbf{x}_j) - \mathbf{r}_j}_{\text{residual gap}} \|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{10}$$

- $\mathbf{p}$ - direction vector
- $\mathbf{r}$ - inexact residual

If additionally

$$\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{11}$$

then the cost function can be monitored as

$$J(\mathbf{x}_j) - J(\mathbf{x}_*) \leq \epsilon J(\mathbf{x}_*)$$

[4] Gratton et al, 2021

# Minimisation: Inexact conjugate gradient[4]

- *Inexact* matrix-vector multiplication (a costly operation!)
- By gradually increasing the inexactness of the mat-vec product, one could still produce the exact version's level of accuracy.
- In our case, the perturbation matrix is

$$E(k) = A - A^* = -(F^N)^T(F^N - P(k))$$

**Theorem in nutshell (a bit modified...)**: At icg-iteration $j$ if

$$\|E_j p_j\|_{A^{-1}} \leq \omega_j \left(\|\mathbf{b}\|_{A^{-1}}, \|\mathbf{p}_j\|_A, \|\mathbf{r}\|_{A^{-1}}\right) \tag{9}$$

then

$$\| \underbrace{\mathbf{r}(\mathbf{x}_j) - \mathbf{r}_j}_{\text{residual gap}} \|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{10}$$

- $\mathbf{p}$ - direction vector
- $\mathbf{r}$ - inexact residual

If additionally

$$\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{11}$$

then the cost function can be monitored as

$$J(\mathbf{x}_j) - J(\mathbf{x}_*) \leq \epsilon J(\mathbf{x}_*)$$

[4] Gratton et al, 2021

# Minimisation: Inexact conjugate gradient[4]

- *Inexact* matrix-vector multiplication (a costly operation!)
- By gradually increasing the inexactness of the mat-vec product, one could still produce the exact version's level of accuracy.
- In our case, the perturbation matrix is

$$E(k) = A - A^* = -(F^N)^T(F^N - P(k))$$

**Theorem in nutshell (a bit modified...)**: At icg-iteration $j$ if

$$\|E_j p_j\|_{A^{-1}} \leq \omega_j \left(\|\mathbf{b}\|_{A^{-1}}, \|\mathbf{p}_j\|_A, \|\mathbf{r}\|_{A^{-1}}\right) \tag{9}$$

then

$$\| \underbrace{\mathbf{r}(\mathbf{x}_j) - \mathbf{r}_j}_{\text{residual gap}} \|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{10}$$

If additionally

$$\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{11}$$

then the cost function can be monitored as

$$J(\mathbf{x}_j) - J(\mathbf{x}_*) \leq \epsilon J(\mathbf{x}_*)$$

- $\mathbf{p}$ - direction vector
- $\mathbf{r}$ - inexact residual

[4] Gratton et al, 2021

# Minimisation: Inexact conjugate gradient[4]

- *Inexact* matrix-vector multiplication (a costly operation!)
- By gradually increasing the inexactness of the mat-vec product, one could still produce the exact version's level of accuracy.
- In our case, the perturbation matrix is

$$E(k) = A - A^* = -(F^N)^T(F^N - P(k))$$

**Theorem in nutshell (a bit modified...)**: At icg-iteration $j$ if

$$\|E_j p_j\|_{A^{-1}} \leq \omega_j \left(\|\mathbf{b}\|_{A^{-1}}, \|\mathbf{p}_j\|_A, \|\mathbf{r}\|_{A^{-1}}\right) \tag{9}$$

then

$$\| \underbrace{\mathbf{r}(\mathbf{x}_j) - \mathbf{r}_j}_{\text{residual gap}} \|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{10}$$

- **p** - direction vector
- **r** - inexact residual

If additionally

$$\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{11}$$

then the cost function can be monitored as

$$J(\mathbf{x}_j) - J(\mathbf{x}_*) \leq \epsilon J(\mathbf{x}_*)$$

[4] Gratton et al, 2021

# Minimisation: Inexact conjugate gradient[4]

- *Inexact* matrix-vector multiplication (a costly operation!)
- By gradually increasing the inexactness of the mat-vec product, one could still produce the exact version's level of accuracy.
- In our case, the perturbation matrix is

$$E(k) = A - A^* = -(F^N)^T(F^N - P(k))$$

**Theorem in nutshell (a bit modified...)**: At icg-iteration $j$ if

$$\|E_j p_j\|_{A^{-1}} \leq \omega_j \left(\|\mathbf{b}\|_{A^{-1}}, \|\mathbf{p}_j\|_A, \|\mathbf{r}\|_{A^{-1}}\right) \tag{9}$$

then

$$\| \underbrace{\mathbf{r}(\mathbf{x}_j) - \mathbf{r}_j}_{\text{residual gap}} \|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{10}$$

- $\mathbf{p}$ - direction vector
- $\mathbf{r}$ - inexact residual

If additionally

$$\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{A^{-1}} \tag{11}$$

then the cost function can be monitored as

$$J(\mathbf{x}_j) - J(\mathbf{x}_*) \leq \epsilon J(\mathbf{x}_*)$$

[4] Gratton et al, 2021

- In real problems $A$ has a huge size $\implies$ impractical estimates.
- Gratton et al approximated $\|\mathbf{p}_j\|_A, J(\mathbf{x}_j), \|\mathbf{b}\|_{A^{-1}}$ and the termination test (11) by easily computable quantities.
  - $\|\mathbf{p}_j\|_A \approx \sqrt{\frac{1}{n}\mathrm{Tr}(A)}\,\|\mathbf{p}_j\|_2$
  - $J(\mathbf{x}_j) \approx J_j \stackrel{\text{def}}{=} -\frac{1}{2} b^T \mathbf{x}_j$
  - $\|\mathbf{b}\|_{A^{-1}} \approx \frac{\|\mathbf{b}\|_2}{\sqrt{\lambda_{\max}}} \quad j = 0,\ \mathbf{x}_0 = 0 \quad \text{and} \quad \|\mathbf{b}\|_{A^{-1}} \approx \sqrt{2|J_j|} \quad j = 1, 2, ..., j_{\max}$
  - $\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\epsilon}{2}\|\mathbf{b}\|_{A^{-1}}$ by $J_{j-d} - J_j \leq \frac{1}{4}\epsilon|J_j|$ for some integer $d$
- But now $\|E_j\mathbf{p}_j\|_{A^{-1}}$ also needs an approximation!

**Result**

If $F$ is invertible, $\|E_j(k)\mathbf{p}_j\|_{A^{-1}}$ and $\|P(k)\mathbf{p}_j - F^N\mathbf{p}_j\|_2$ are rigorously the same whatever the parareal approximation. $F^N\mathbf{p}_j$ is the corresponding exact solution.

- In real problems $A$ has a huge size $\implies$ impractical estimates.
- Gratton et al approximated $\|\mathbf{p}_j\|_A, J(\mathbf{x}_j), \|\mathbf{b}\|_{A^{-1}}$ and the termination test (11) by easily computable quantities.
  - $\|\mathbf{p}_j\|_A \approx \sqrt{\frac{1}{n}\mathrm{Tr}(A)}\,\|\mathbf{p}_j\|_2$
  - $J(\mathbf{x}_j) \approx J_j \stackrel{\text{def}}{=} -\frac{1}{2} b^T \mathbf{x}_j$
  - $\|\mathbf{b}\|_{A^{-1}} \approx \frac{\|\mathbf{b}\|_2}{\sqrt{\lambda_{\max}}} \quad j = 0,\ \mathbf{x}_0 = 0 \quad$ and $\quad \|\mathbf{b}\|_{A^{-1}} \approx \sqrt{2|J_j|} \quad j = 1, 2, ..., j_{\max}$
  - $\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\epsilon}{2}\|\mathbf{b}\|_{A^{-1}}$ by $J_{j-d} - J_j \leq \frac{1}{4}\epsilon|J_j|$ for some integer $d$
- But now $\|E_j\mathbf{p}_j\|_{A^{-1}}$ also needs an approximation!

**Result**

If $F$ is invertible, $\|E_j(k)\mathbf{p}_j\|_{A^{-1}}$ and $\|P(k)\mathbf{p}_j - F^N\mathbf{p}_j\|_2$ are rigorously the same whatever the parareal approximation. $F^N\mathbf{p}_j$ is the corresponding exact solution.

- In real problems $A$ has a huge size $\implies$ impractical estimates.
- Gratton et al approximated $\|\mathbf{p}_j\|_A, J(\mathbf{x}_j), \|\mathbf{b}\|_{A^{-1}}$ and the termination test (11) by easily computable quantities.
  - $\|\mathbf{p}_j\|_A \approx \sqrt{\frac{1}{n}\mathrm{Tr}(A)}\,\|\mathbf{p}_j\|_2$
  - $J(\mathbf{x}_j) \approx J_j \stackrel{\text{def}}{=} -\frac{1}{2} b^T \mathbf{x}_j$
  - $\|\mathbf{b}\|_{A^{-1}} \approx \frac{\|\mathbf{b}\|_2}{\sqrt{\lambda_{\max}}} \quad j = 0,\ \mathbf{x}_0 = 0 \quad \text{and} \quad \|\mathbf{b}\|_{A^{-1}} \approx \sqrt{2|J_j|} \quad j = 1, 2, ..., j_{\max}$
  - $\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\epsilon}{2}\|\mathbf{b}\|_{A^{-1}}$ by $J_{j-d} - J_j \leq \frac{1}{4}\epsilon|J_j|$ for some integer $d$
- But now $\|E_j\mathbf{p}_j\|_{A^{-1}}$ also needs an approximation!

**Result**

If $F$ is invertible, $\|E_j(k)\mathbf{p}_j\|_{A^{-1}}$ and $\|P(k)\mathbf{p}_j - F^N\mathbf{p}_j\|_2$ are rigorously the same whatever the parareal approximation. $F^N\mathbf{p}_j$ is the corresponding exact solution.

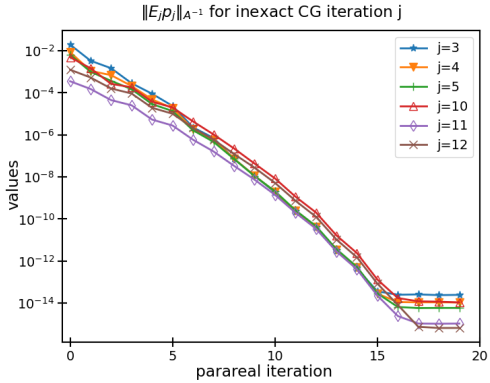
- In real problems $A$ has a huge size $\implies$ impractical estimates.
- Gratton et al approximated $\|\mathbf{p}_j\|_A, J(\mathbf{x}_j), \|\mathbf{b}\|_{A^{-1}}$ and the termination test (11) by easily computable quantities.
  - $\|\mathbf{p}_j\|_A \approx \sqrt{\frac{1}{n}\mathrm{Tr}(A)}\,\|\mathbf{p}_j\|_2$
  - $J(\mathbf{x}_j) \approx J_j \stackrel{\text{def}}{=} -\frac{1}{2} b^T \mathbf{x}_j$
  - $\|\mathbf{b}\|_{A^{-1}} \approx \frac{\|\mathbf{b}\|_2}{\sqrt{\lambda_{\max}}} \quad j = 0,\ \mathbf{x}_0 = 0 \quad \text{and} \quad \|\mathbf{b}\|_{A^{-1}} \approx \sqrt{2|J_j|} \quad j = 1, 2, ..., j_{\max}$
  - $\|\mathbf{r}_j\|_{A^{-1}} \leq \frac{\epsilon}{2}\|\mathbf{b}\|_{A^{-1}}$ by $J_{j-d} - J_j \leq \frac{1}{4}\epsilon|J_j|$ for some integer $d$
- But now $\|E_j\mathbf{p}_j\|_{A^{-1}}$ also needs an approximation!

**Result**

If $F$ is invertible, $\|E_j(k)\mathbf{p}_j\|_{A^{-1}}$ and $\|P(k)\mathbf{p}_j - F^N\mathbf{p}_j\|_2$ are rigorously the same whatever the parareal approximation. $F^N\mathbf{p}_j$ is the corresponding exact solution.

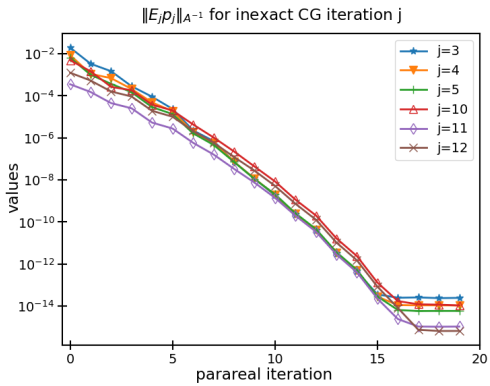
Well now we also need an approximation for $F^N \mathbf{p}_j$ !

Well now we also need an approximation for $F^N \mathbf{p}_j$ !



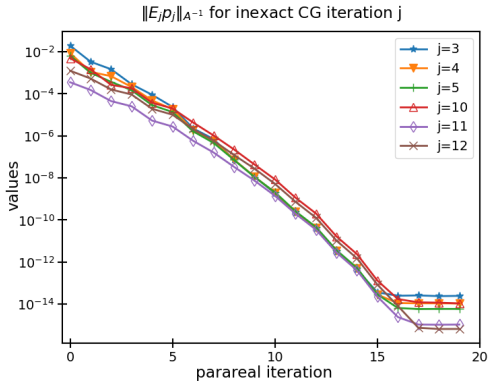

- similar profile for $\|E_j \mathbf{p}_j\|_{A^{-1}}$

Well now we also need an approximation for $F^N \mathbf{p}_j$ !




- similar profile for $\|E_j \mathbf{p}_j\|_{A^{-1}}$
- Start with parareal with a reasonable tolerance and then use previous parareal iterates to approximate $\|\mathbf{p}_j\|$ for subsequent icg-iterations

Well now we also need an approximation for $F^N \mathbf{p}_j$ !



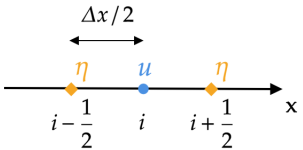

- similar profile for $\|E_j \mathbf{p}_j\|_{A^{-1}}$
- Start with parareal with a reasonable tolerance and then use previous parareal iterates to approximate $\|\mathbf{p}_j\|$ for subsequent icg-iterations
- Check that $\xi_j$ doesn't lie between last two norm values

1d linear shallow water model

$$\frac{\partial \eta}{\partial t} = -H\frac{\partial u}{\partial x}$$
$$\frac{\partial u}{\partial t} = -g\frac{\partial \eta}{\partial x} + \mu\frac{\partial^2 u}{\partial x^2} \tag{12}$$

**Setup:** Arakawa C-grid discretisation

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = C\mathbf{x}, \qquad \mathbf{x} = \begin{pmatrix}\eta \\ u\end{pmatrix}$$




Parameters

- $N = 20$
- no. of fine time steps per time window, $N_f = 100$
- no. of coarse time steps per time window, $N_g = 20$
- coarse time step, $\Delta t = 0.05$
- spatial grid-step, $\Delta x = 1$
- diffusion constant, $\mu = 0.15$
- average height, $H = 0.9$

Coarse and fine propagators are defined using theta scheme ($\theta = 0.51$)

$$F = [(I - \theta\delta t C)^{-1}(I + (1-\theta)\delta t C)]^{N_f}$$

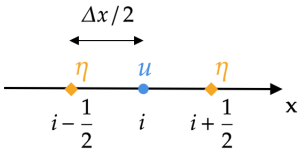
$$G = [(I - \theta\Delta t C)^{-1}(I + (1-\theta)\Delta t C)]^{N_g}$$

1d linear shallow water model

$$\frac{\partial \eta}{\partial t} = -H\frac{\partial u}{\partial x}$$
$$\frac{\partial u}{\partial t} = -g\frac{\partial \eta}{\partial x} + \mu\frac{\partial^2 u}{\partial x^2} \tag{12}$$

**Setup:** Arakawa C-grid discretisation

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = C\mathbf{x}, \qquad \mathbf{x} = \begin{pmatrix}\eta \\ u\end{pmatrix}$$




**Parameters**

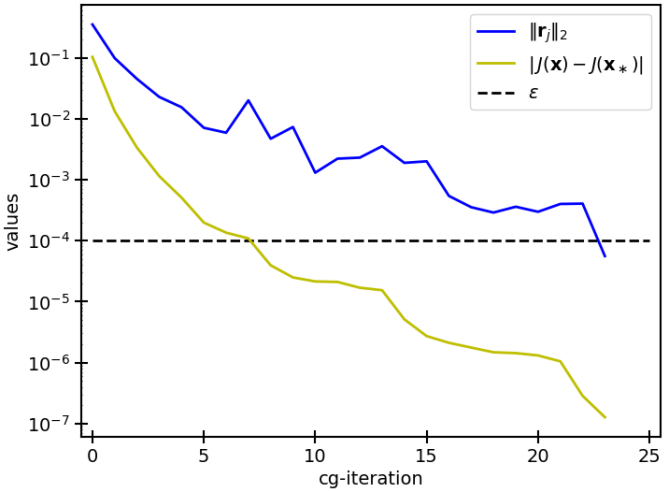
- $N = 20$
- no. of fine time steps per time window, $N_f = 100$
- no. of coarse time steps per time window, $N_g = 20$
- coarse time step, $\Delta t = 0.05$
- spatial grid-step, $\Delta x = 1$
- diffusion constant, $\mu = 0.15$
- average height, $H = 0.9$

Coarse and fine propagators are defined using theta scheme ($\theta = 0.51$)

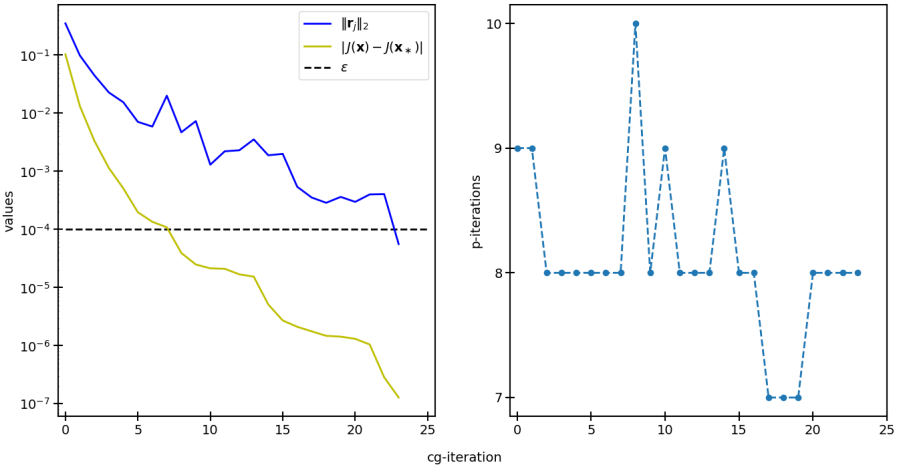
$$F = [(I - \theta\delta tC)^{-1}(I + (1-\theta)\delta tC)]^{N_f}$$

$$G = [(I - \theta\Delta tC)^{-1}(I + (1-\theta)\Delta tC)]^{N_g}$$

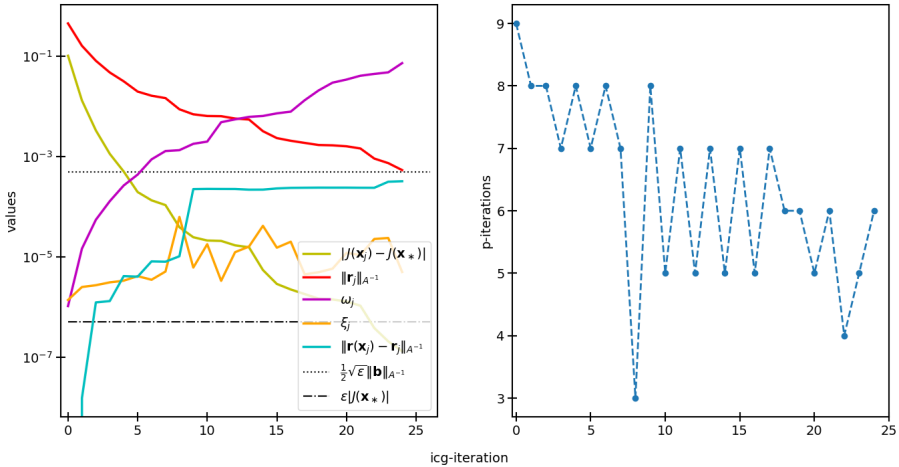
- Initial condition - a gaussian at the middle of the basin/domain
- Using exact CG with $\epsilon_{cg} = 10^{-4}$, takes 24 minimisation iterations



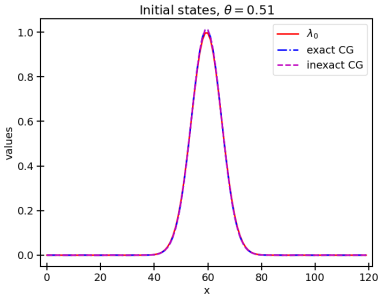
- Fixed stopping tolerance for parareal $\epsilon_{\mathrm{p}} = 10^{-6},\ \epsilon_{\mathrm{cg}} = 10^{-4}$
- On average 8.125 p-iterations per icg-iteration

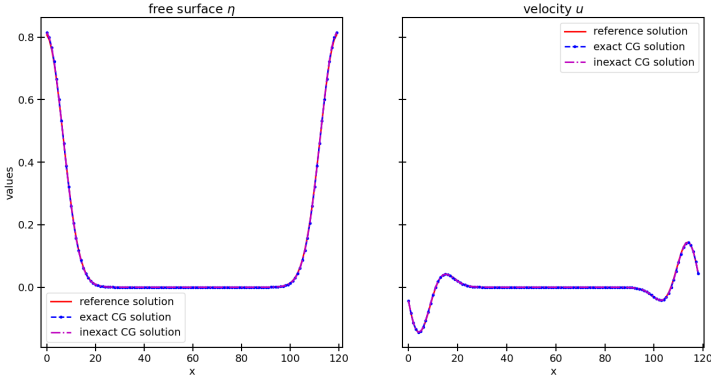


- On average 6.36 p-iterations per icg-iteration, nice!
- Theoretical speedup: $\frac{N}{\text{p-iterations}} = \frac{20}{6.36} = 3.1$
- $\omega_j$ is the tolerance for the error norm $\|E_j(k)\|_{A^{-1},A}$



# Retreived initial condition and final results



- Reducing the cost of the coarse solver.
  - Using the Krylov enhanced parareal[5] version. Choosing the right vectors for the convergence of Krylov enhanced method because of storage issues.
- Running a parallel version of adjoint - running parareal but in reverse direction leading to double degree of parallelism. Similar error analysis for adjoint.
- Current processes work on synchronicity. Introduce asynchronous methods which don't wait for neighbouring processes and are more linked to the spatial discretisation.
- Other ways to couple DA and PinT algorithms - solving the forward and backward model together as a system as done in ParaOpt[6].
- Extension to 2d model

[5] Gander and Petcu, 2008
[6] Gander, Kwok and Salomon, 2020