Probabilistic Linear Solvers

Jon Cockayne, Chris Oates, Ilse Ipsen, Mark Girolami

June 21, 2019

In This Talk...

We will construct probabilistic numerical methods for solving linear systems.

Solving Linear Systems

The Problem

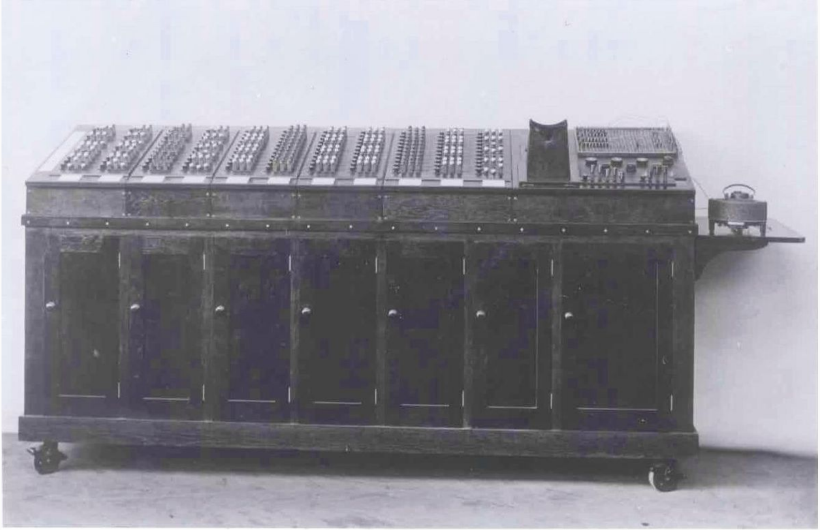
Goal: find $\boldsymbol{x}^*$ in

$$A\boldsymbol{x}^* = \boldsymbol{b}$$

$A \in \mathbb{R}^{d \times d}$ invertible (not necessarily SPD).

$\boldsymbol{x}^*, \boldsymbol{b} \in \mathbb{R}^d$.

The First Algorithm Ever Implemented?



“Mallock Machine”, capable of solving 6×6 linear systems.

Direct Methods

Direct Methods aim to solve the system "in one shot".

Direct Methods

Direct Methods aim to solve the system "in one shot".

E.g. Cholesky factorisation:

1. Compute $A = LL^\top$

Direct Methods

Direct Methods aim to solve the system "in one shot".

E.g. Cholesky factorisation:

1. Compute $A = LL^\top$
2. Solve $L\boldsymbol{z} = \boldsymbol{b}$.

Direct Methods

Direct Methods aim to solve the system “in one shot”.

E.g. Cholesky factorisation:

1. Compute $A = LL^\top$
2. Solve $L\boldsymbol{z} = \boldsymbol{b}$.
3. Solve $L^\top \boldsymbol{x} = \boldsymbol{z}$

Direct Methods

Direct Methods aim to solve the system "in one shot".

E.g. Cholesky factorisation:

1. Compute $A = LL^\top$
2. Solve $L\boldsymbol{z} = \boldsymbol{b}$.
3. Solve $L^\top \boldsymbol{x} = \boldsymbol{z}$

(Naive) cost: $\mathcal{O}(d^3)$ computation, $\mathcal{O}(d^2)$ storage.

Iterative Methods

Iterative Methods aim to produce a sequence $(\boldsymbol{x}_m) \to \boldsymbol{x}^*$ as $m \to \infty$.

Iterative Methods

Iterative Methods aim to produce a sequence $(\boldsymbol{x}_m) \to \boldsymbol{x}^*$ as $m \to \infty$.

Often possible to elicit an iterative method that is faster than a direct method if we are willing to accept a small error in the result.

Iterative Methods

Iterative Methods aim to produce a sequence $(\boldsymbol{x}_m) \to \boldsymbol{x}^*$ as $m \to \infty$.

Often possible to elicit an iterative method that is faster than a direct method if we are willing to accept a small error in the result.

Generally require some "initial guess" $\boldsymbol{x}_0$; then

$$\boldsymbol{x}_m = P_m(\boldsymbol{x}_0; \boldsymbol{x}^*)$$

The Conjugate Gradient Method[1]

A non-stationary, non-linear iterative method.

[1] Hestenes and Stiefel [1952]

The Conjugate Gradient Method[1]

A non-stationary, non-linear iterative method.

Consider the functional:

$$f(\boldsymbol{x}) := \boldsymbol{x}^\top A\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$$

Has a unique minimum $\boldsymbol{x}^*$.

[1] Hestenes and Stiefel [1952]

The Conjugate Gradient Method[1]

A non-stationary, non-linear iterative method.

Consider the functional:

$$f(\boldsymbol{x}) := \boldsymbol{x}^\top A\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$$

Has a unique minimum $\boldsymbol{x}^*$.

CG arises from performing modified gradient descent on this functional to find its minimum.

[1] Hestenes and Stiefel [1952]

The Conjugate Gradient Method

Raw gradient descent:

$$\boldsymbol{s}_m = \boldsymbol{b} - A\boldsymbol{x}_m = \boldsymbol{r}_m$$

CG search directions:

$$\boldsymbol{s}_m = \boldsymbol{r}_m - \langle \boldsymbol{r}_m, \boldsymbol{s}_{m-1} \rangle_A \cdot \boldsymbol{s}_{m-1}$$

The Conjugate Gradient Method

Raw gradient descent:

$$\boldsymbol{s}_m = \boldsymbol{b} - A\boldsymbol{x}_m = \boldsymbol{r}_m$$

CG search directions:

$$\boldsymbol{s}_m = \boldsymbol{r}_m - \langle \boldsymbol{r}_m, \boldsymbol{s}_{m-1} \rangle_A \cdot \boldsymbol{s}_{m-1}$$

Produces a set of search directions that are A-orthonormal (after normalisation):

$$\langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle_A = \delta_{ij}$$

Computational Cost

- $\mathcal{O}(md^2)$ computation (1 matrix-vector multiplication per-iteration).

Computational Cost

- $\mathcal{O}(md^2)$ computation (1 matrix-vector multiplication per-iteration).
- $\mathcal{O}(d)$ storage (need to store 2-3 additional vectors).

Classical Theory

Introduce the Krylov Subspace:

$$K_m(A, \boldsymbol{b}) = \mathsf{span}(\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{m-1}\boldsymbol{b})$$

Classical Theory

Introduce the Krylov Subspace:

$$K_m(A, \boldsymbol{b}) = \mathsf{span}(\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{m-1}\boldsymbol{b})$$

Theorem (Krylov Subspace Method)

We have that

$$\boldsymbol{x}_m = \underset{\boldsymbol{x} \in \boldsymbol{x}_0 + K_m(A, \boldsymbol{r}_0)}{\arg\min} \|\boldsymbol{x} - \boldsymbol{x}^*\|_A$$

Introduce the Krylov Subspace:

$$K_m(A, \boldsymbol{b}) = \mathsf{span}(\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{m-1}\boldsymbol{b})$$

Theorem (Krylov Subspace Method)

We have that

$$\boldsymbol{x}_m = \underset{\boldsymbol{x} \in \boldsymbol{x}_0 + K_m(A, \boldsymbol{r}_0)}{\arg\min} \|\boldsymbol{x} - \boldsymbol{x}^*\|_A$$

Theorem (Convergence)

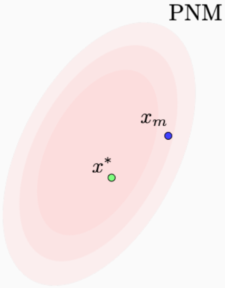
We have that

$$\frac{\|\boldsymbol{x}_m - \boldsymbol{x}^*\|_A}{\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|_A} \leq 2\left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right)^m$$

Probabilistic Numerical Methods

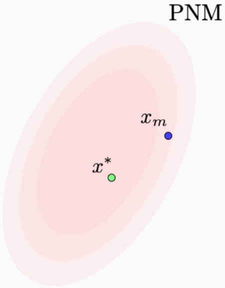
Probabilistic Numerical Methods

Numerical methods that return probability measures.



Numerical methods that return probability measures.

Those measures are designed to describe where the truth might lie given the computational effort expended.

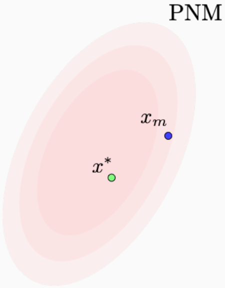


Probabilistic Numerical Methods

Numerical methods that return probability measures.

Those measures are designed to describe where the truth might lie given the computational effort expended.

Methods are called Bayesian if the output is a posterior [Cockayne et al., 2019].



Why Use PNM?

- Contemporary numerical problems involve composition of many base numerical methods into pipelines.

Why Use PNM?

- Contemporary numerical problems involve composition of many base numerical methods into pipelines.
- BPNM can be straightforwardly composed under mild conditions Cockayne et al. [2019].

BayesCG

Probabilistic Linear Solvers

- Start with a Gaussian prior

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}_0, \Sigma_0)$$

Probabilistic Linear Solvers

- Start with a Gaussian prior

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}_0, \Sigma_0)$$

- Use the data provided by some set of search directions to construct the posterior:

$$\boldsymbol{s}_m^\top A \boldsymbol{x}^* = \boldsymbol{s}_m^\top \boldsymbol{b} := y_m$$

Probabilistic Linear Solvers

- Start with a Gaussian prior

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}_0, \Sigma_0)$$

- Use the data provided by some set of search directions to construct the posterior:

$$\boldsymbol{s}_m^\top A \boldsymbol{x}^* = \boldsymbol{s}_m^\top \boldsymbol{b} := y_m$$

Let

$$S_m = \begin{pmatrix} \boldsymbol{s}_1 & \cdots & \boldsymbol{s}_m \end{pmatrix}$$

$$
\begin{aligned}
\boldsymbol{x}|\boldsymbol{y}_m &\sim \mathcal{N}(\boldsymbol{x}_m, \Sigma_m) \\
\boldsymbol{x}_m &= \boldsymbol{x}_0 + \Sigma_0 A^\top S_m \Lambda_m^{-1} S_m^\top (\boldsymbol{b} - A\boldsymbol{x}_0) \\
\Sigma_m &= \Sigma_0 - \Sigma_0 A^\top S_m \Lambda_m^{-1} S_m^\top A \Sigma_0
\end{aligned}
$$

$$
\Lambda_m = S_m^\top A \Sigma_0 A^\top S_m
$$

A Problem

To compute the posterior we must invert

$$\Lambda_m = S_m^\top A \Sigma_0 A^\top S_m$$

A Problem

To compute the posterior we must invert

$$\Lambda_m = S_m^\top A \Sigma_0 A^\top S_m$$

However

$$(\Lambda_m)_{ij} = \langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle_{A\Sigma_0 A^\top}$$

Choosing $A\Sigma_0 A^\top$-orthonormal search directions makes this more practical.

Theorem (BayesCG)

Let

$$\tilde{\boldsymbol{s}}_m = \boldsymbol{r}_{m-1} - \langle \boldsymbol{s}_{m-1}, \boldsymbol{r}_{m-1} \rangle_{A\Sigma_0 A^\top} \cdot \boldsymbol{s}_{m-1}$$

Then after normalisation the directions $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m$ *are* $A\Sigma_0 A^\top$*-orthonormal.*

Theorem (BayesCG)

Let

$$\tilde{\boldsymbol{s}}_m = \boldsymbol{r}_{m-1} - \langle \boldsymbol{s}_{m-1}, \boldsymbol{r}_{m-1} \rangle_{A\Sigma_0 A^\top} \cdot \boldsymbol{s}_{m-1}$$

Then after normalisation the directions $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m$ *are* $A\Sigma_0 A^\top$*-orthonormal.*

Furthermore we have

$$\boldsymbol{x}_m = \boldsymbol{x}_{m-1} + \Sigma_0 A^\top \boldsymbol{s}_m (\boldsymbol{s}_m^\top \boldsymbol{r}_{m-1})$$
$$\Sigma_m = \Sigma_{m-1} - \Sigma_0 A^\top \boldsymbol{s}_m \boldsymbol{s}_m^\top A \Sigma_0$$

Cost

- $\mathcal{O}(md^2)$ computation. (2-3 matrix-vector multiplications per-iter).
- $\mathcal{O}(md)$ storage (need to store search directions).

Cost

- $\mathcal{O}(md^2)$ computation. (2-3 matrix-vector multiplications per-iter).
- $\mathcal{O}(md)$ storage (need to store search directions).

More costly than CG, but comes with UQ.

Theoretical Results I

Theorem (Krylov Subspace Method)

Let

$$K_m^* = \boldsymbol{x}_0 + \Sigma_0 A^\top K_m(A\Sigma_0 A^\top, \boldsymbol{r}_0)$$

Theoretical Results I

Theorem (Krylov Subspace Method)

Let

$$K_m^* = \boldsymbol{x}_0 + \Sigma_0 A^\top K_m(A\Sigma_0 A^\top, \boldsymbol{r}_0)$$

Then the BayesCG posterior mean satisfies

$$\boldsymbol{x}_m = \underset{\boldsymbol{x} \in K_m^*}{\arg\min} \|\boldsymbol{x} - \boldsymbol{x}^*\|_{\Sigma_0^{-1}}$$

Theoretical Results I

Theorem (Krylov Subspace Method)

Let

$$K_m^* = \boldsymbol{x}_0 + \Sigma_0 A^\top K_m(A\Sigma_0 A^\top, \boldsymbol{r}_0)$$

Then the BayesCG posterior mean satisfies

$$\boldsymbol{x}_m = \underset{\boldsymbol{x} \in K_m^*}{\arg\min} \|\boldsymbol{x} - \boldsymbol{x}^*\|_{\Sigma_0^{-1}}$$

Note that $\Sigma_0 = A^{-1}$ replicates CG!

Theoretical Results II

Theorem (Convergence Rate)

$$\frac{\|\boldsymbol{x}_m - \boldsymbol{x}^*\|_{\Sigma_0^{-1}}}{\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|_{\Sigma_0^{-1}}} \leq 2\left(\frac{\sqrt{\kappa(\Sigma_0 A^\top A)} - 1}{\sqrt{\kappa(\Sigma_0 A^\top A)} + 1}\right)^m$$

Theoretical Results II

Theorem (Convergence Rate)

$$\frac{\|\boldsymbol{x}_m - \boldsymbol{x}^*\|_{\Sigma_0^{-1}}}{\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|_{\Sigma_0^{-1}}} \leq 2\left(\frac{\sqrt{\kappa(\Sigma_0 A^\top A)} - 1}{\sqrt{\kappa(\Sigma_0 A^\top A)} + 1}\right)^m$$

Fastest convergence achieved when $\kappa(\Sigma_0 A^\top A) \approx 1$.

Experimental Results

Priors Considered

- $\Sigma_0 = A^{-1}$: Replicates CG.

Priors Considered

- $\Sigma_0 = A^{-1}$: Replicates CG.
- $\Sigma_0 = I$: "Uninformative".

Priors Considered

- $\Sigma_0 = A^{-1}$: Replicates CG.
- $\Sigma_0 = I$: "Uninformative".
- A-Priori Optimal Directions: Essentially random.

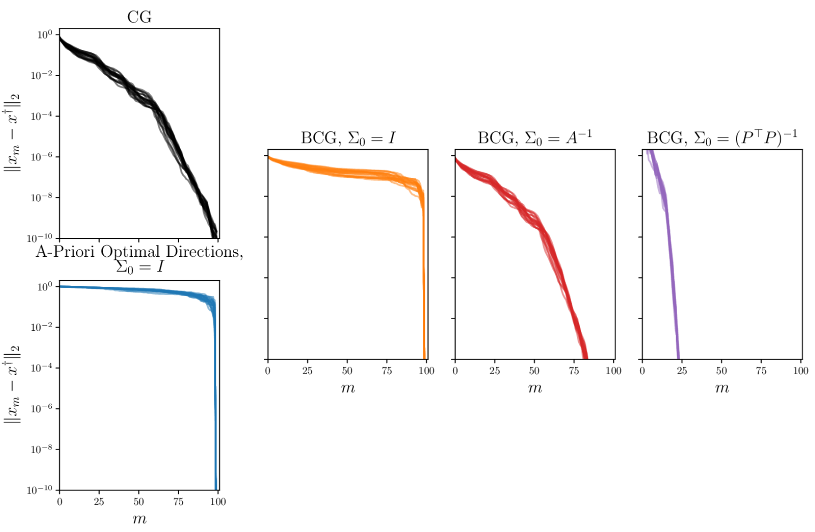
Priors Considered

- $\Sigma_0 = A^{-1}$: Replicates CG.
- $\Sigma_0 = I$: "Uninformative".
- A-Priori Optimal Directions: Essentially random.
- Preconditioner Prior: Given a preconditioner P for A, set $\Sigma_0 = (P^\top P)^{-1}$.

Experimental Setup

- A a random sparse matrix (drawn using the matlab function `sprandsym`).
- $d = 100$.
- Many test problems $\boldsymbol{x}^*$ are drawn from $\mathcal{N}(\mathbf{0}, I)$.
- BayesCG applied to $m = 100$.

Convergence of Posterior Mean



Assessment of Posterior UQ

To assess the UQ we make the ansatz that if the posterior is "well-calibrated" then $\boldsymbol{x}^*$ should look like a draw from the posterior.

Assessment of Posterior UQ

To assess the UQ we make the ansatz that if the posterior is "well-calibrated" then $\boldsymbol{x}^*$ should look like a draw from the posterior.

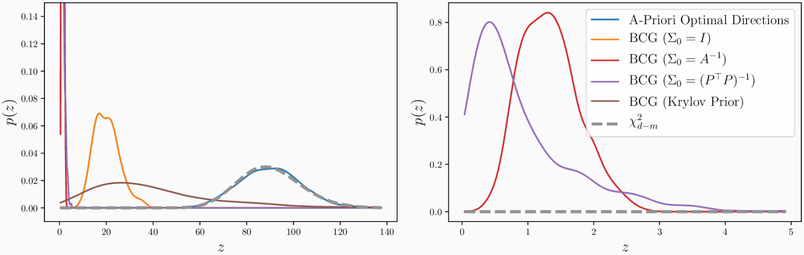
Then for the Z-statistic:

$$Z(\boldsymbol{x}^*) := \|\boldsymbol{x}^* - \boldsymbol{x}_m\|^2_{\Sigma_m^\dagger}$$

we can prove that under the ansatz:

$$Z(\boldsymbol{x}^*) \sim \chi^2_{d-m}$$

Assessment of Posterior UQ



Poor UQ

$$S_m^\top(\boldsymbol{x}^*) A \boldsymbol{x} = S_m^\top(\boldsymbol{x}^*) A \boldsymbol{x}^*$$

Non-Bayesian Methods

Stationary Iterative Methods[2]

Iteration is of the form

$$P_m = \underbrace{P \circ \cdots \circ P}_{m \text{ times}}$$

i.e. each iteration is independent of all previous iterations.

[2] Young [1971]

Stationary Iterative Methods[2]

Iteration is of the form

$$P_m = \underbrace{P \circ \cdots \circ P}_{m \text{ times}}$$

i.e. each iteration is independent of all previous iterations.

In stationary iterative methods of first order:

$$P(\boldsymbol{x}) := G\boldsymbol{x} + \boldsymbol{f}$$
$$G \in \mathbb{R}^{d \times d}$$
$$f \in \mathbb{R}^{d}$$

[2] Young [1971]

Stationary Iterative Methods[2]

Iteration is of the form

$$P_m = \underbrace{P \circ \cdots \circ P}_{m \text{ times}}$$

i.e. each iteration is independent of all previous iterations.

In stationary iterative methods of first order:

$$P(\boldsymbol{x}) := G\boldsymbol{x} + \boldsymbol{f}$$
$$G \in \mathbb{R}^{d \times d}$$
$$f \in \mathbb{R}^{d}$$

Examples: Jacobi iteration, Richardson iteration, ...

[2] Young [1971]

Pushforward Methods

Stationary iterative methods do not obviously give rise to a Bayesian approach.

Pushforward Methods

Stationary iterative methods do not obviously give rise to a Bayesian approach.

Bayesian methods are generally more expensive than classical methods (often much more).

Pushforward Methods

For an iterative method P_m define the associated pushforward method:

$$\mu_m = (P_m)_{\#}\mu_0$$

where $P_{\#}\mu$ is defined as

$$[P_{\#}\mu](B) = \mu(P^{-1}B)$$

Pushforward Methods

For an iterative method P_m define the associated pushforward method:

$$\mu_m = (P_m)_{\#}\mu_0$$

where $P_{\#}\mu$ is defined as

$$[P_{\#}\mu](B) = \mu(P^{-1}B)$$

Accessible via a simple sampling algorithm:

1. Draw $\boldsymbol{x} \sim \mu_0$
2. Compute $P_m(\boldsymbol{x})$

Pushforward Stationary Iterative Methods

Theorem (Probabilistic Linear Stationary Iterative Method of First Degree)

Suppose $\mu_0 \sim \mathcal{N}(\boldsymbol{x}_0, \Sigma_0)$ *and*

$$P_m = \underbrace{P \circ \cdots \circ P}_{m \text{ times}}$$

with $P(\boldsymbol{x}) = G\boldsymbol{x} + \boldsymbol{f}$. *Then*

$$\mu_m = \mathcal{N}(\boldsymbol{x}_m, \Sigma_m)$$

$$\boldsymbol{x}_m = G^m \boldsymbol{x}_0 + \sum_{i=1}^{m-1} G^{m-i} \boldsymbol{f}$$

$$\Sigma_m = G^m \Sigma_0 (G^m)^\top$$

But Why?

Assess these methods using the Z-statistic:

$$Z(\boldsymbol{x}^*) = \|\boldsymbol{x}^* - \boldsymbol{x}_m\|^2_{\Sigma_m^\dagger}$$

But Why?

Assess these methods using the Z-statistic:

$$Z(\boldsymbol{x}^*) = \|\boldsymbol{x}^* - \boldsymbol{x}_m\|^2_{\Sigma_m^\dagger}$$

Theorem

Suppose Σ_0 is full-rank and G is a diagonalisable matrix of rank r. Then $\text{rank}(\Sigma_m) = r$ and

$$Z(\boldsymbol{x}^*) \sim \chi^2_r.$$

But Why?

Assess these methods using the Z-statistic:

$$Z(\boldsymbol{x}^*) = \|\boldsymbol{x}^* - \boldsymbol{x}_m\|^2_{\Sigma_m^\dagger}$$

Theorem

Suppose Σ_0 is full-rank and G is a diagonalisable matrix of rank r. Then $\operatorname{rank}(\Sigma_m) = r$ and

$$Z(\boldsymbol{x}^*) \sim \chi^2_r.$$

Thus these methods are automatically well-calibrated.

Non-linear-Gaussian Case

The S-statistic is defined as

$$S(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2.$$

Non-linear-Gaussian Case

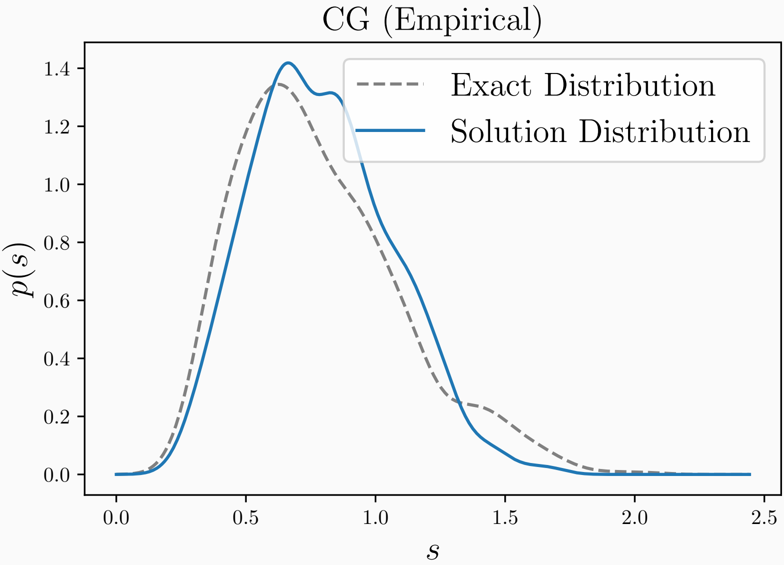
The S-statistic is defined as

$$S(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2.$$

Let $X^* \sim \mu_{\text{ref}}$ and $X, X' \sim \mu_m$ i.i.d. Then we say μ_m is well-calibrated wrt μ_{ref} if

$$S(X, X') = S(X, X^*)$$

Calibration of Pushforward CG



Conclusions

Current Developments: BayesCG

- Stability properties in finite-precision.

Current Developments: BayesCG

- Stability properties in finite-precision.
- Accelerating convergence while obtaining better UQ:
  - Further work on the Krylov prior.
  - "Pushforward" methods.

Current Developments: BayesCG

- Stability properties in finite-precision.
- Accelerating convergence while obtaining better UQ:
  - Further work on the Krylov prior.
  - “Pushforward” methods.

Discussion now open!

Current Developments: Pushforward Methods

- Further theory - generalising “well-calibrated”.

Current Developments: Pushforward Methods

- Further theory - generalising "well-calibrated".
- Applications to other methods than linear systems?
  - Optimizers?
  - Eigenproblems?
  - ...?

Questions?

References

Jon Cockayne, Chris Oates, Tim Sullivan, and Mark Girolami. Bayesian probabilistic numerical methods. *SIAM Review*, 2019. to appear.

M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409, December 1952. doi: 10.6028/jres.049.044. URL `https://doi.org/10.6028/jres.049.044`.

David M. Young. *Iterative Solution of Large Linear Systems*. Elsevier, 1971. doi: 10.1016/c2013-0-11733-3. URL `https://doi.org/10.1016/c2013-0-11733-3`.