Assimilation of nonlinear observations using the maximum likelihood ensemble filter with exact Newton optimization

Takeshi Enomoto and Saori Nakashita
Kyoto University

The authors thank Prof. Milija Zupanski for helpful discussions.

19 October 2023, 9th International Symposium on Data Assimilation, Palazzo dei Congressi, Bologna, Italia

科研費 KAKENHI This work was supported by KAKENHI Grant Number JP19H05605, JP21K03662 and JP22KJ1966.

Pavillon de l'Esprit Nouveau

Qualitative classification of the linearity

Chapter 5, Rodgers (2000)

- *Linear*: when the forward model can be put in the form of $\mathbf{y} = \mathbf{K}\mathbf{x}$ and any a priori is Gaussian; very few practical problems are truly linear.
- *Nearly linear*: problems which are non-linear, but for which a linearisation about some prior state is adequate to find a solution.
- *Moderately non-linear*: problems where linearisation is adequate for the error analysis, but not for finding a solution. Many problems are of this kind.
- *Grossly non-linear*: problems which are non-linear even within the range of the errors.

If the problems is not too non-linear, Newton iteration is a straightforward numerical method for finding the zero of the gradient of the the cost function J.

Questions

The exact Newton (EN) and conjugate gradient (CG) methods are compared using the maximum likelihood ensemble filter (MLEF, Zupanski 2005).

1. How does EN or CG minimize benchmark functions?
2. How does an ensemble approximation of the gradient and Hessian matrix affect optimization with EN or CG?
3. How does EN or CG perform in cycled experiments with a simple model?

Nonlinear least-square problems

Similarities between benchmark and cost functions

- Booth function: $f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$
- Rosenbrock function: $f(x, y) = (1 - x)^2 + 100(y - x^2)^2$
- Cost function:

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}^{\mathrm{f}})^{\mathrm{T}}\mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}^{\mathrm{f}}) + \frac{1}{2}\left[\mathbf{y} - H(\mathbf{x})\right]^{\mathrm{T}}\mathbf{R}^{-1}\left[\mathbf{y} - H(\mathbf{x})\right]$$

- Nonlinear least-square problems: $f(\mathbf{x}) = \frac{1}{2}\sum_{i}^{m}[f_i(\mathbf{x})]^2$

The exact Newton method

The Hessian matrix is used to solve the Newton equation.

1. Compute the gradient $\mathbf{g}_k = \nabla f_k$ and Hessian matrix $\mathbf{G}_k = \nabla^2 f_k$ for a quadratic approximation with a descent vector $\mathbf{d}_k = \mathbf{x} - \mathbf{x}_k$:
   $$f(\mathbf{x}_k + \mathbf{d}_k) \approx f(\mathbf{x}_k) + \mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k + \frac{1}{2}\mathbf{d}_k^{\mathrm{T}}\mathbf{G}_k\mathbf{d}_{\mathrm{k}}$$

2. Solve the Newton equation $\mathbf{G}_k\mathbf{d}_k = -\mathbf{g}_k$.

3. Update the state: $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k\mathbf{G}_k^{-1}\mathbf{g}_k$.
   The exact Newton (EN) method uses a unit step size $\alpha_k = 1$

The Gauss–Newton method

The gradient and Hessian matrix is approximated with the Jacobian matrix.

1. Compute the Jacobian matrix $\mathbf{F} = \dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}$ of the residual vector $\mathbf{f} = \begin{pmatrix} f_1(\mathbf{x}) & f_2(\mathbf{x}) & \dots & f_m(\mathbf{x}) \end{pmatrix}^{\mathrm{T}}$.

2. Approximate the gradient $\mathbf{g} = \mathbf{F}^{\mathrm{T}}\mathbf{f}$ and Hessian matrix $\mathbf{G} = \mathbf{F}^{\mathrm{T}}\mathbf{F}$.

3. Solve the Newton equation for the descent vector $\mathbf{d} = -\mathbf{G}^{-1}\mathbf{g} = -(\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{f}$.

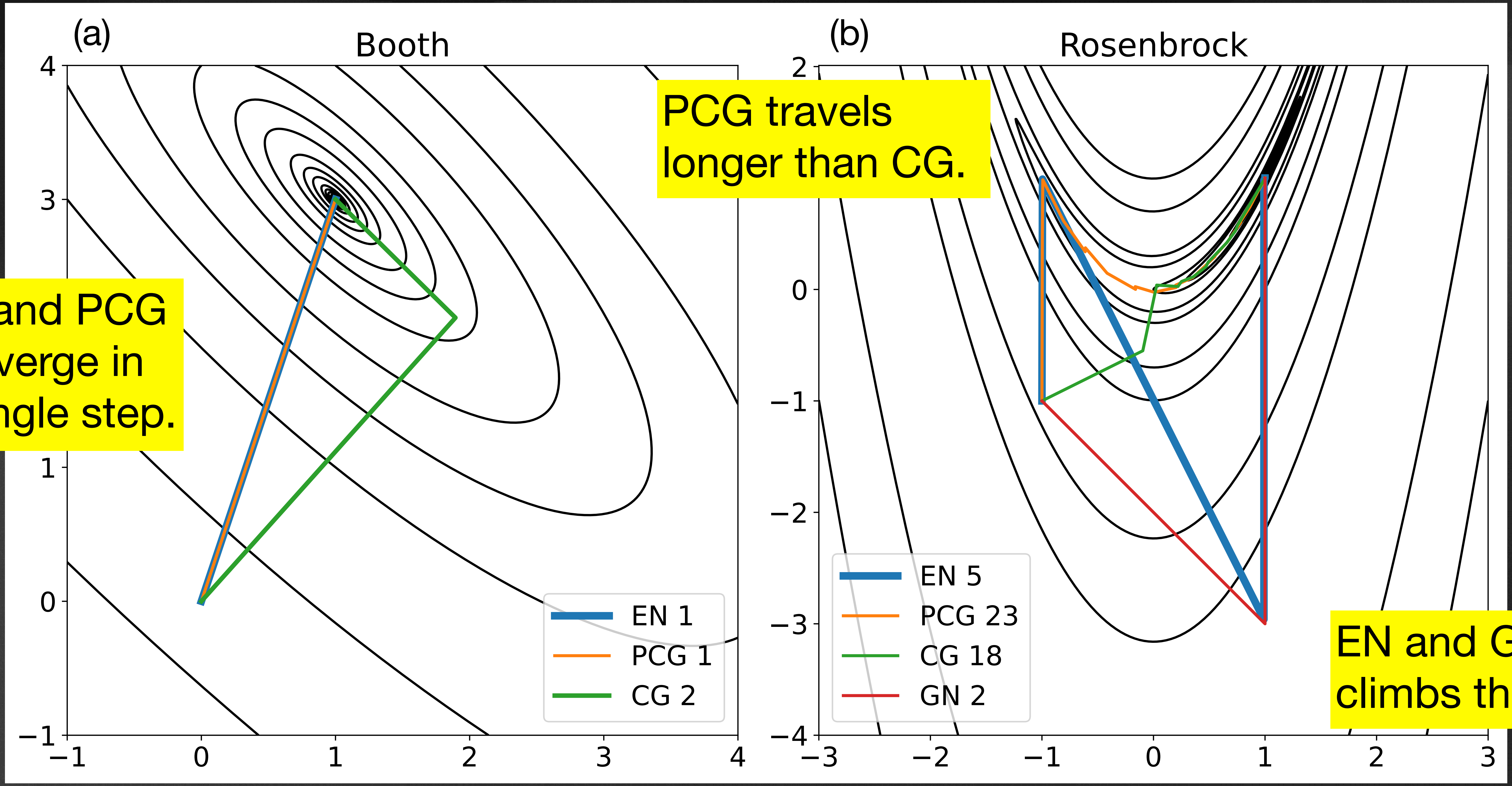
The conjugate gradient method

1. Initialize the descent direction $\mathbf{d}_0 = -\mathbf{g}_0$ where $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$.
2. Compute a step length α_k with $f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) = \min_\alpha f(\mathbf{x}_k + \alpha \mathbf{d}_k)$.
3. Update the state with $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.
4. Update the descent direction with $\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_{k+1}\mathbf{d}_k$ where $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$ and $\beta_{k+1} = \max\left[0, \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^T \mathbf{g}_k}\right]$.

The Hessian matrix can be used for preconditioning $\mathbf{x}' = \mathbf{G}^{-\mathrm{T}/2}\mathbf{x}$.

Optimization of benchmark functions

$$f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2 \qquad f(x, y) = (1 - x)^2 + 100(y - x^2)^2$$



The maximum likelihood ensemble filter

Zupanski (2005)

- Cost function $J(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}^{\mathrm{f}})^{\mathrm{T}}\mathbf{P}_{\mathrm{f}}^{-1}(\mathbf{x} - \mathbf{x}^{\mathrm{f}}) + \frac{1}{2}\left[\mathbf{y} - H(\mathbf{x})\right]^{\mathrm{T}}\mathbf{R}^{-1}\left[\mathbf{y} - H(\mathbf{x})\right]$

  where $\mathbf{x}$ is the state, $\mathbf{x}^{\mathrm{f}}$ is the control forecast, $\mathbf{y}$ is the observation and $\mathbf{B}$ and $\mathbf{R}$ are forecast and observation error covariance matrices, respectively.

- Each column of the square root of $\mathbf{P}_{\mathrm{f}}$ is forecast perturbations.

$$\mathbf{P}_{\mathrm{f}}^{1/2} = \left[\mathbf{p}_1^{\mathrm{f}} \quad \mathbf{p}_2^{\mathrm{f}} \quad \cdots \quad \mathbf{p}_k^{\mathrm{f}}\right], \mathbf{p}_j^{\mathrm{f}} = M(\mathbf{x}^{\mathrm{a}} + \mathbf{p}_j^{\mathrm{a}}) - M(\mathbf{x}^{\mathrm{a}}) = \mathbf{x}_j^{\mathrm{f}} - \mathbf{x}^{\mathrm{f}}, \mathbf{p}_j^{\mathrm{a}} = \mathbf{x}_j^{\mathrm{a}} - \mathbf{x}^{\mathrm{a}}$$

- Represent the departure by a linear combination $\mathbf{x} - \mathbf{x}^{\mathrm{f}} = \mathbf{P}_{\mathrm{f}}^{1/2}\mathbf{w}$.

- Cost function $J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \frac{1}{2}\left[\mathbf{y} - H(\mathbf{x})\right]^{\mathrm{T}}\mathbf{R}^{-1}\left[\mathbf{y} - H(\mathbf{x})\right]$

Hessian preconditioning

- $\mathbf{x} - \mathbf{x}^{\mathrm{f}} = \mathbf{P}_{\mathrm{f}}^{1/2}(\mathbf{I} + \mathbf{C})^{-\mathrm{T}/2}\zeta$ where $\mathbf{C} = \mathbf{Z}^{\mathrm{T}}\mathbf{Z}$ and $\mathbf{I} + \mathbf{C}$ is the Hessian matrix.
- $\mathbf{Z} = \mathbf{R}^{-1/2}\mathbf{H}\mathbf{P}_{\mathrm{f}}^{1/2}$ where $\mathbf{H} = \partial H/\partial \mathbf{x}$ is the Jacobian matrix.
- Alternatively $\mathbf{Z} = [\mathbf{z}_1 \quad \mathbf{z}_2 \quad \cdots \quad \mathbf{z}_k]$, $\mathbf{z}_j = \mathbf{R}^{-1/2}\left[H(\mathbf{x}^{\mathrm{f}} + \mathbf{p}_j^{\mathrm{f}}) - H(\mathbf{x}^{\mathrm{f}})\right]$
- $J(\zeta) = \frac{1}{2}\zeta^{\mathrm{T}}(\mathbf{I} + \mathbf{C})^{-1}\zeta + \frac{1}{2}\left[\mathbf{y} - H(\mathbf{x})\right]^{\mathrm{T}}\mathbf{R}^{-1}\left[\mathbf{y} - H(\mathbf{x})\right]$
- $\nabla_{\zeta} J = (\mathbf{I} + \mathbf{C})^{-1}\zeta - (\mathbf{I} + \mathbf{C})^{-1/2}\mathbf{Z}^{\mathrm{T}}\mathbf{R}^{-1/2}\left[\mathbf{y} - H(\mathbf{x})\right]$
- Update ensemble with $\mathbf{P}_{\mathrm{a}}^{1/2} = \mathbf{P}_{\mathrm{f}}^{1/2}\left(\mathbf{I} + \mathbf{C}(\mathbf{x}^{\mathrm{a}})\right)^{-\mathrm{T}/2}$.

The exact Newton method in MLEF

A proposed method

- $J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \frac{1}{2}\left[\mathbf{y} - H(\mathbf{x})\right]^{\mathrm{T}}\mathbf{R}^{-1}\left[\mathbf{y} - H(\mathbf{x})\right]$
- Approximate $J(\mathbf{w}+\mathbf{d}) \approx J(\mathbf{w}) + \nabla J\mathbf{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}}\nabla^2 J\mathbf{d}$
- Solve the Newton equation $\nabla^2 J\mathbf{d} = -\nabla J$ without computing the inverse, where $\nabla J = \mathbf{w} - \mathbf{Y}^{\mathrm{T}}\mathbf{R}^{-1}\left[\mathbf{y} - H(\mathbf{x})\right], \nabla^2 J = \mathbf{I} + \mathbf{Y}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{Y}$ and $\mathbf{Y} = \mathbf{H}\mathbf{P}_{\mathrm{f}}^{1/2}$ or $\mathbf{Y} = [\mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_k]$ with $\mathbf{y}_j = H(\mathbf{x}^{\mathrm{f}} + \mathbf{p}_j^{\mathrm{f}}) - H(\mathbf{x}^{\mathrm{f}})$.

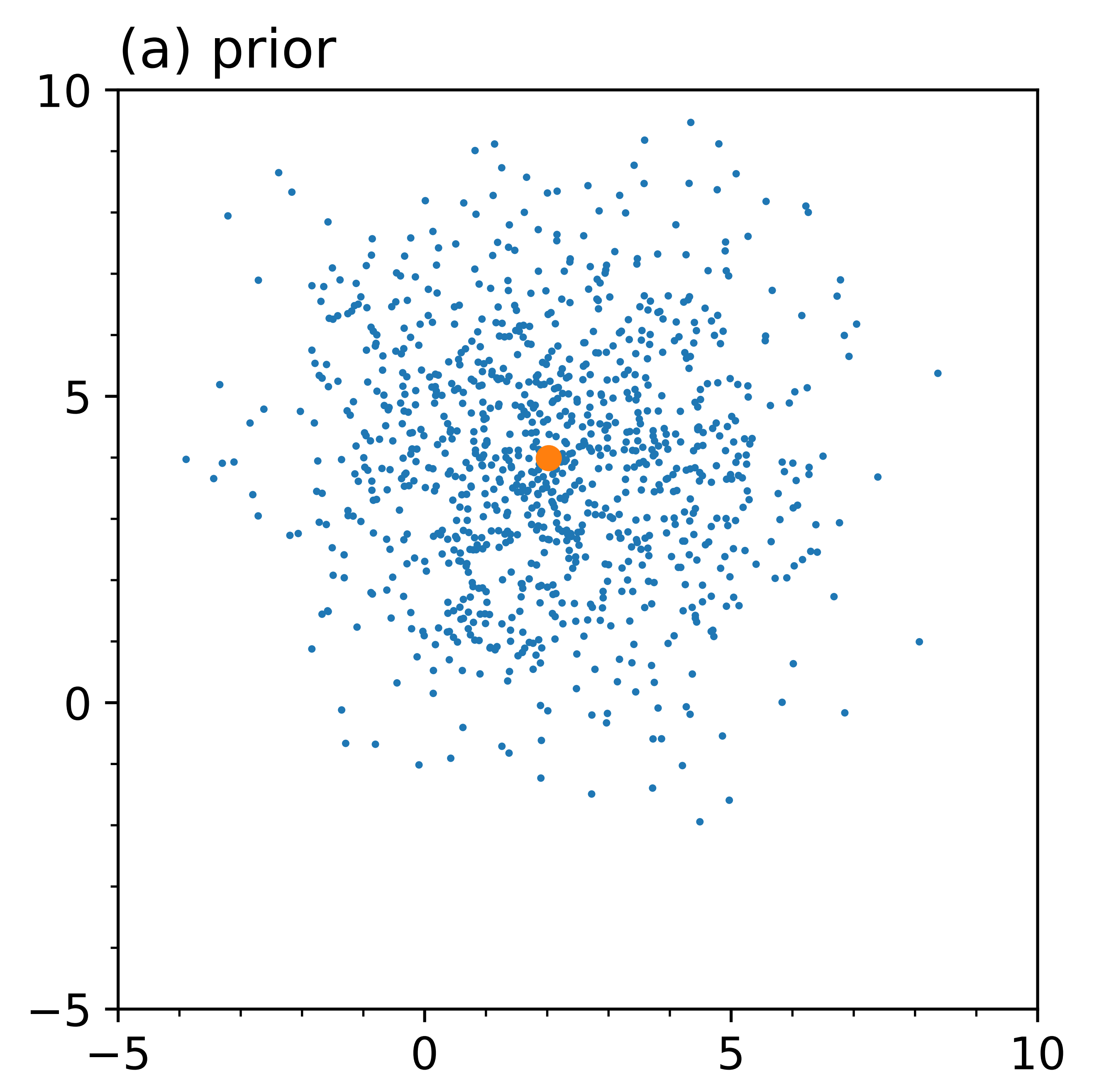
How does MLEF with EN differ from the original?

- The cost function is minimized for the ensemble weight $\mathbf{w}$.
- The Newton equation is solved exactly, avoiding a line search subproblem.
- The Hessian matrix and its inverse is not explicitly computed.
- Unnormalized observation perturbation matrix $\mathbf{Y}$ instead of normalized $\mathbf{Z}$.
- The square root of inverse of $\mathbf{R}$ is not used.

Assimilation of a single wind speed observation

Experimental settings

- Lorenc (2003), Bowler et al. (2013)
- The prior ensemble: 1000 members around $(2, 4)\,\mathrm{m\,s^{-1}}$ with a standard deviation of $(2, 2)\,\mathrm{m\,s^{-1}}$
- Observation: a single wind speed $3\,\mathrm{m\,s^{-1}}$ with a 10% Gaussian error
- Observation operator: $|\mathbf{u}| = \sqrt{u^2 + v^2}$
- Ensemble gradient and Hessian with EN, CG (fixed $\mathbf{Z}$) and CGZ (updated $\mathbf{Z}$).
- Jacobian $\mathbf{H} = \mathbf{u}/|\mathbf{u}|$ with ENJ and CGJ.



Assimilation of a single wind speed observation

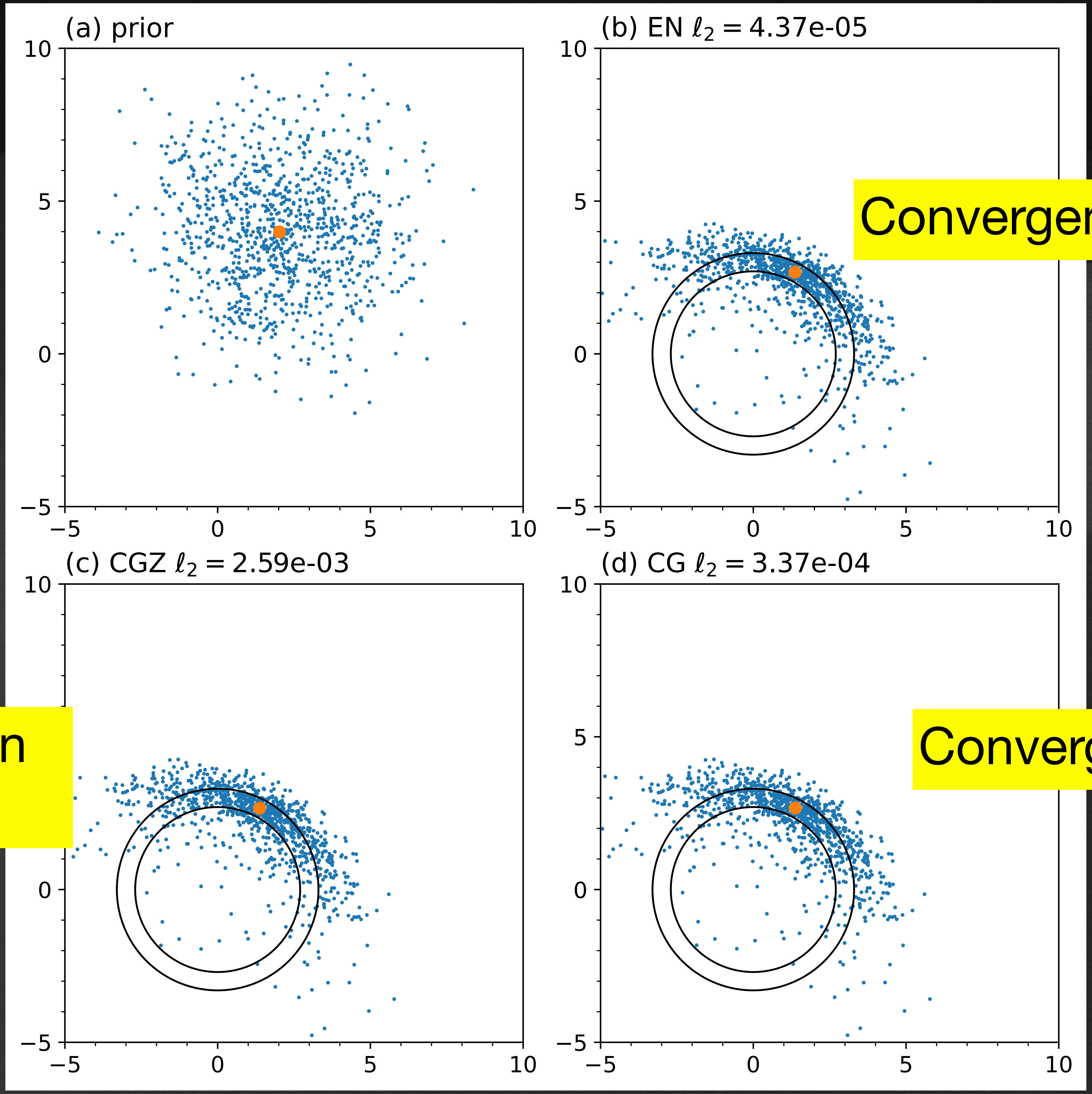
Prior and posterior ensembles

Prior

EN

CGZ

CG



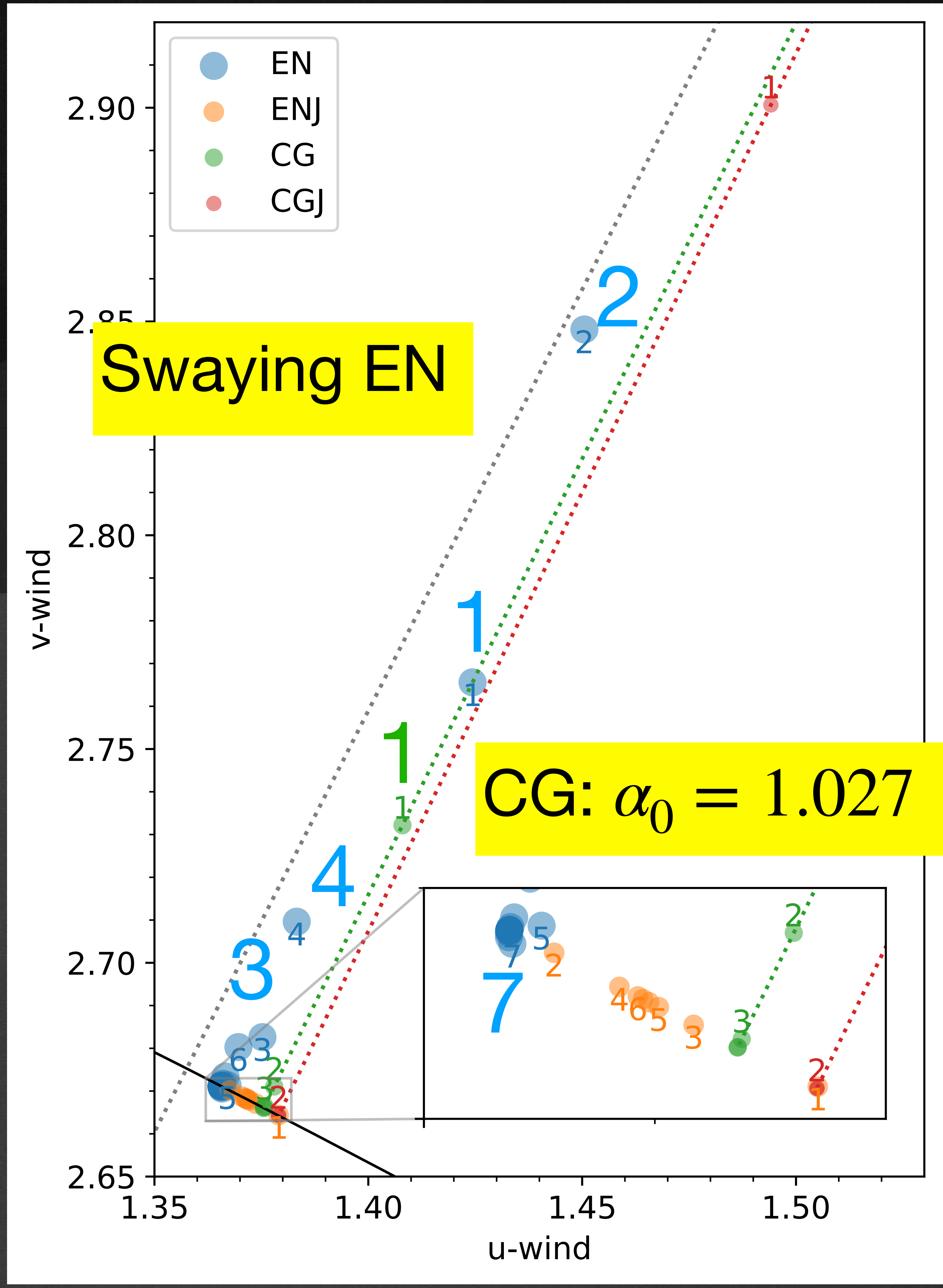
Convergence in 26 steps.

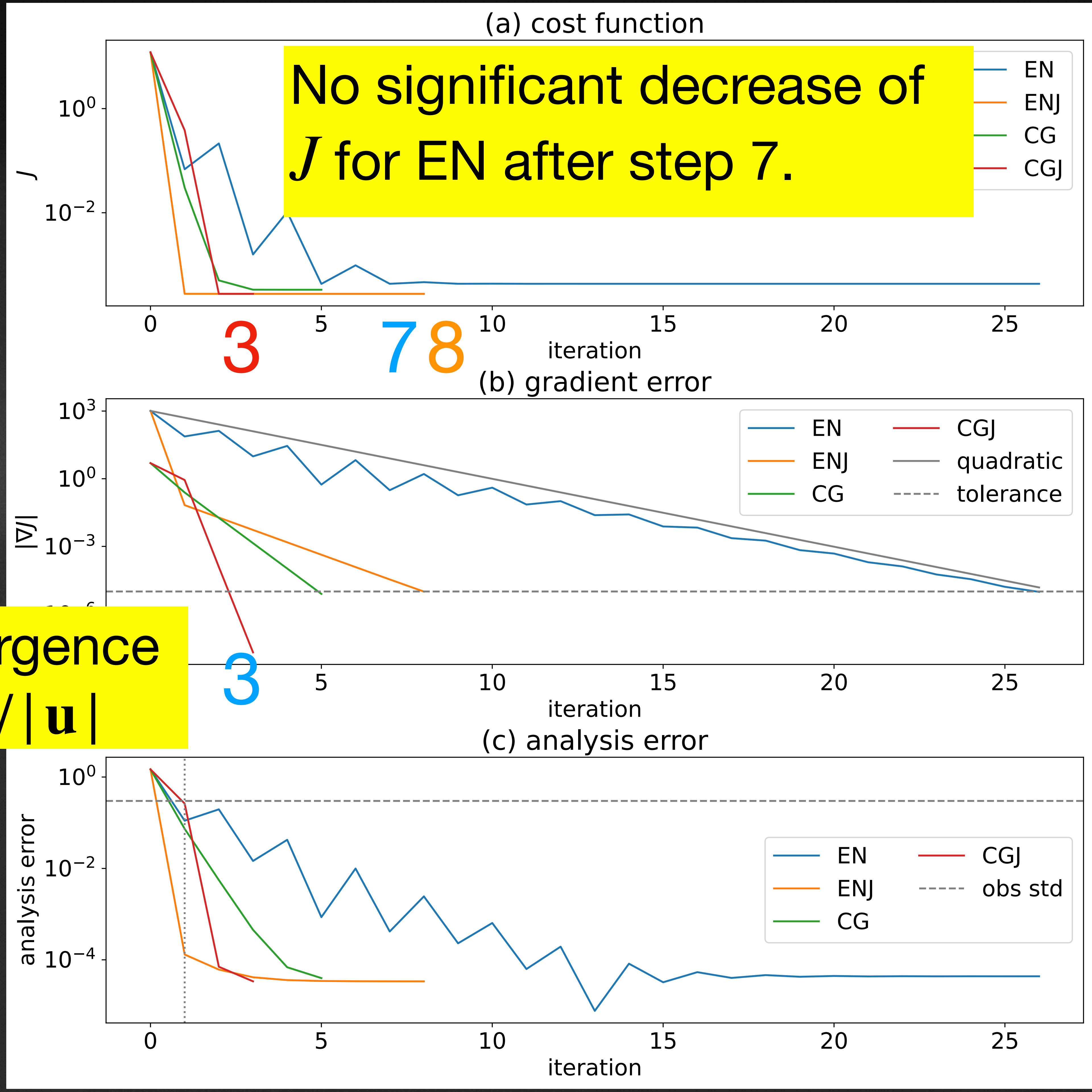
Stagnation at the first iteration due to a line search failure.

Converges in 5 steps.

Assimilation of a single wind speed observation

Optimization history





Fast convergence with $\mathbf{H} = \mathbf{u}/|\mathbf{u}|$

J

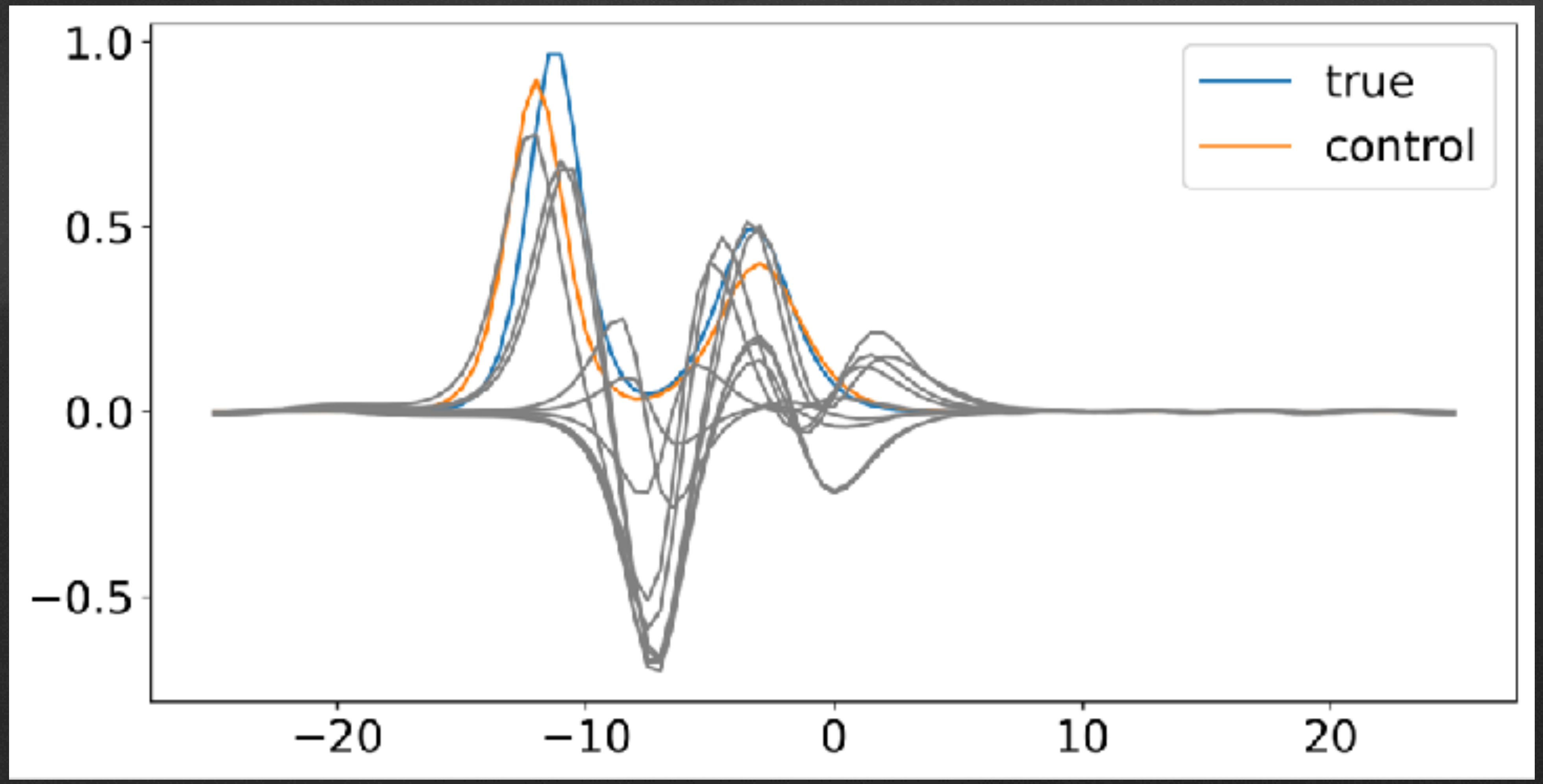
$|\nabla J|$

ℓ_2

Cycled experiments

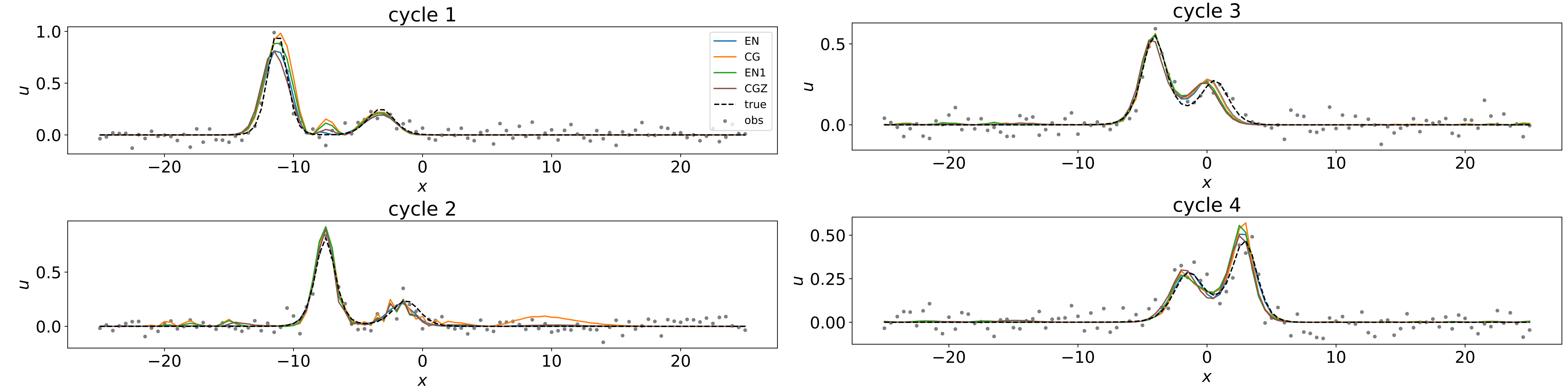
with a Kortweg–de Vries–Burgers (KdVB) model

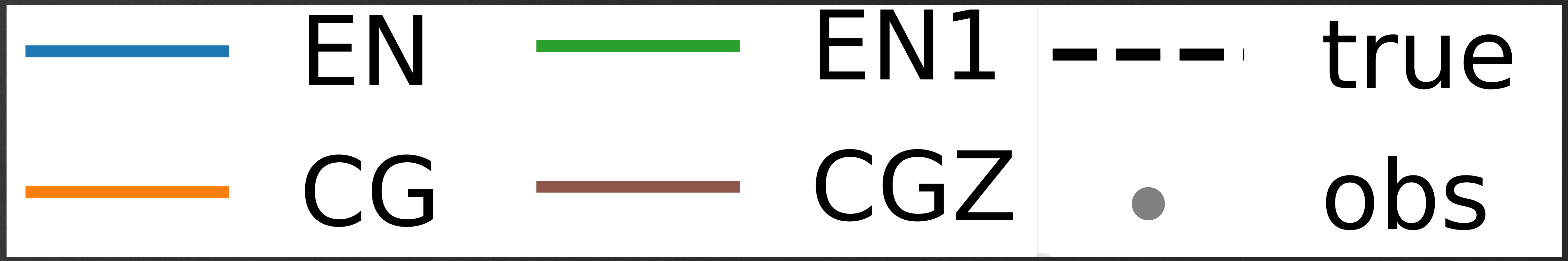
- The KdVB equation $\frac{\partial u}{\partial t} + 6u\frac{\partial u}{\partial x} + \frac{\partial^3 u}{\partial x^3} = \nu\frac{\partial u^2}{\partial x^2}$
- A cyclic domain discretized with $n = 101$ points.
- The true and control runs are integrated from different initial time t_0 of two-soliton solutions with different Bäcklund parameters $\beta_{1,2}$.
- Initial ensemble perturbations are generated from an ensemble forecast with perturbed $\beta_{1,2}$ and t_0.
- Quadratic observations $H(u) = u^2$ are generated by perturbing the true run.



Cycled experiments with a KdVB model

Initial four cycles

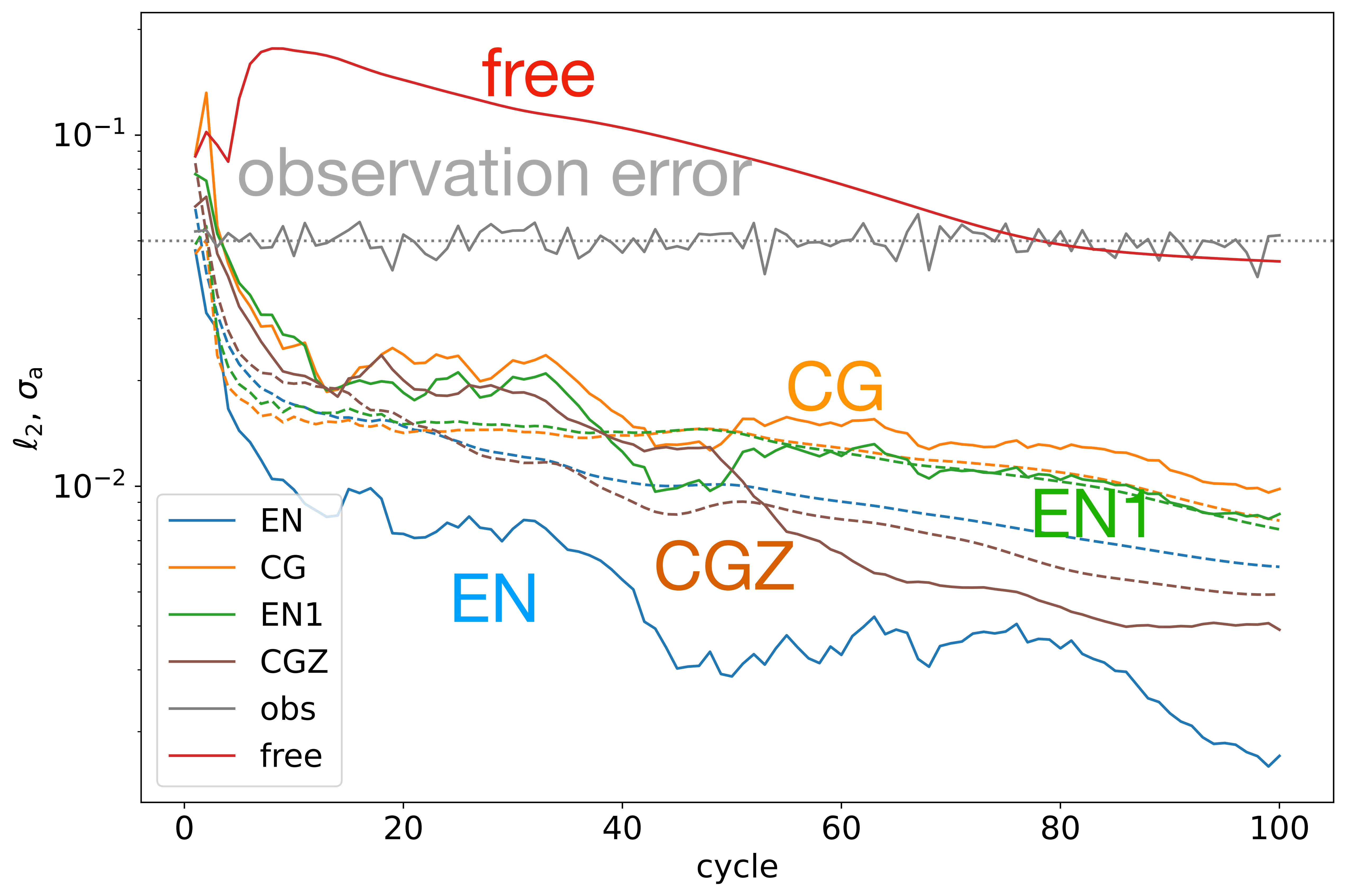




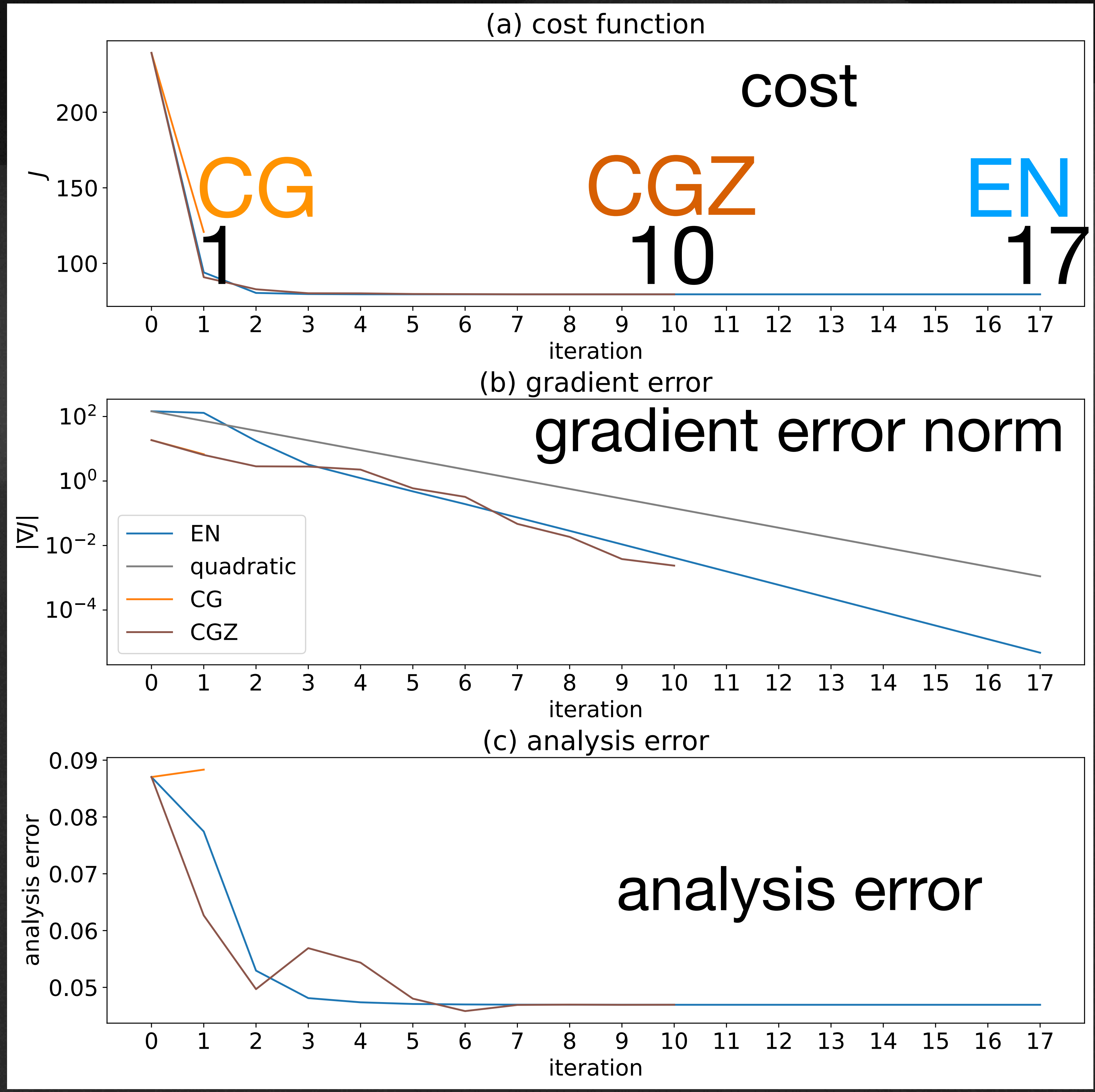
Cycled experiments with a KdVB model

Analysis quality

Analysis error and spread



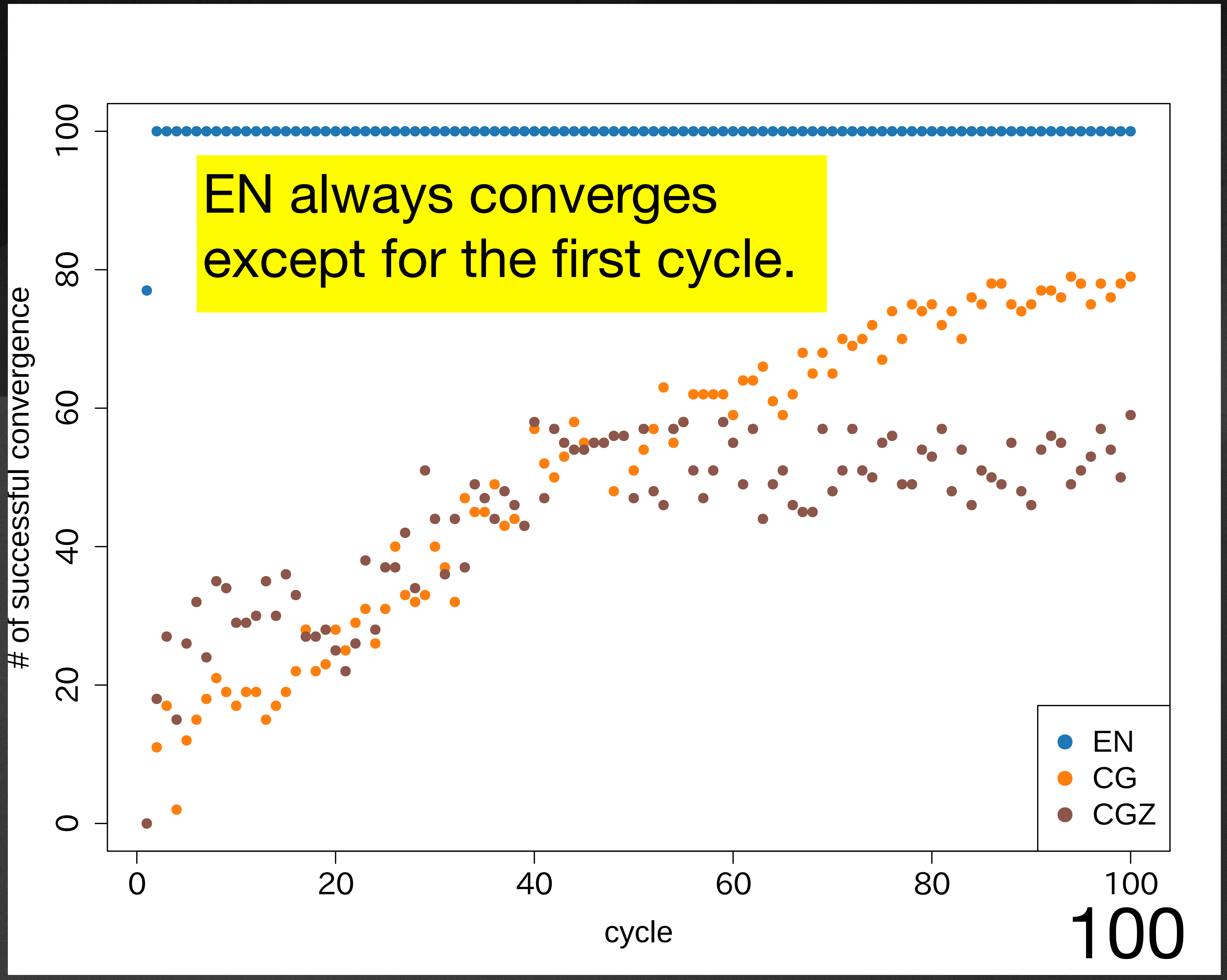
Iterations at the first cycle



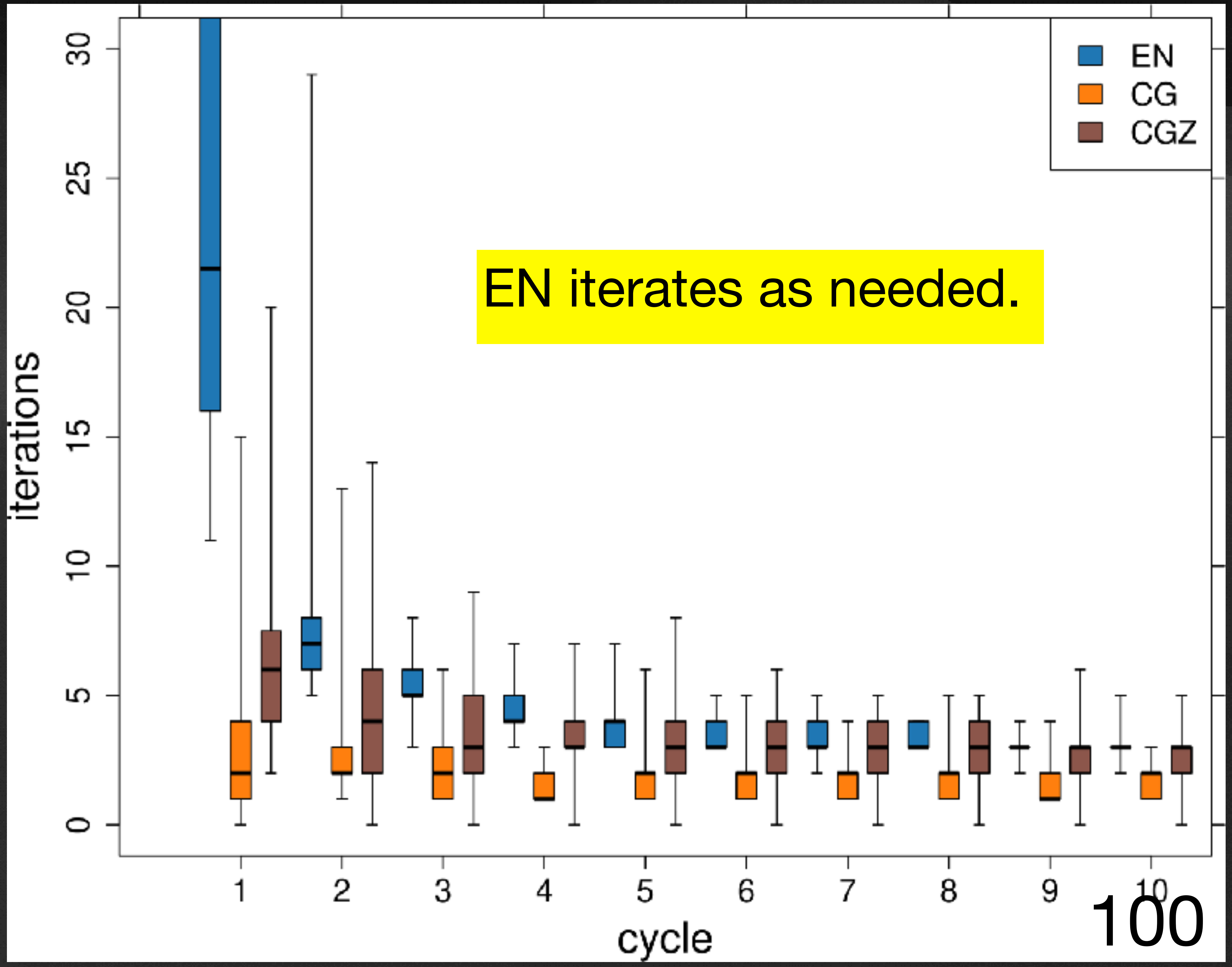
Cycled experiments with a KdVB model

Repeated experiments

Number of successful convergence



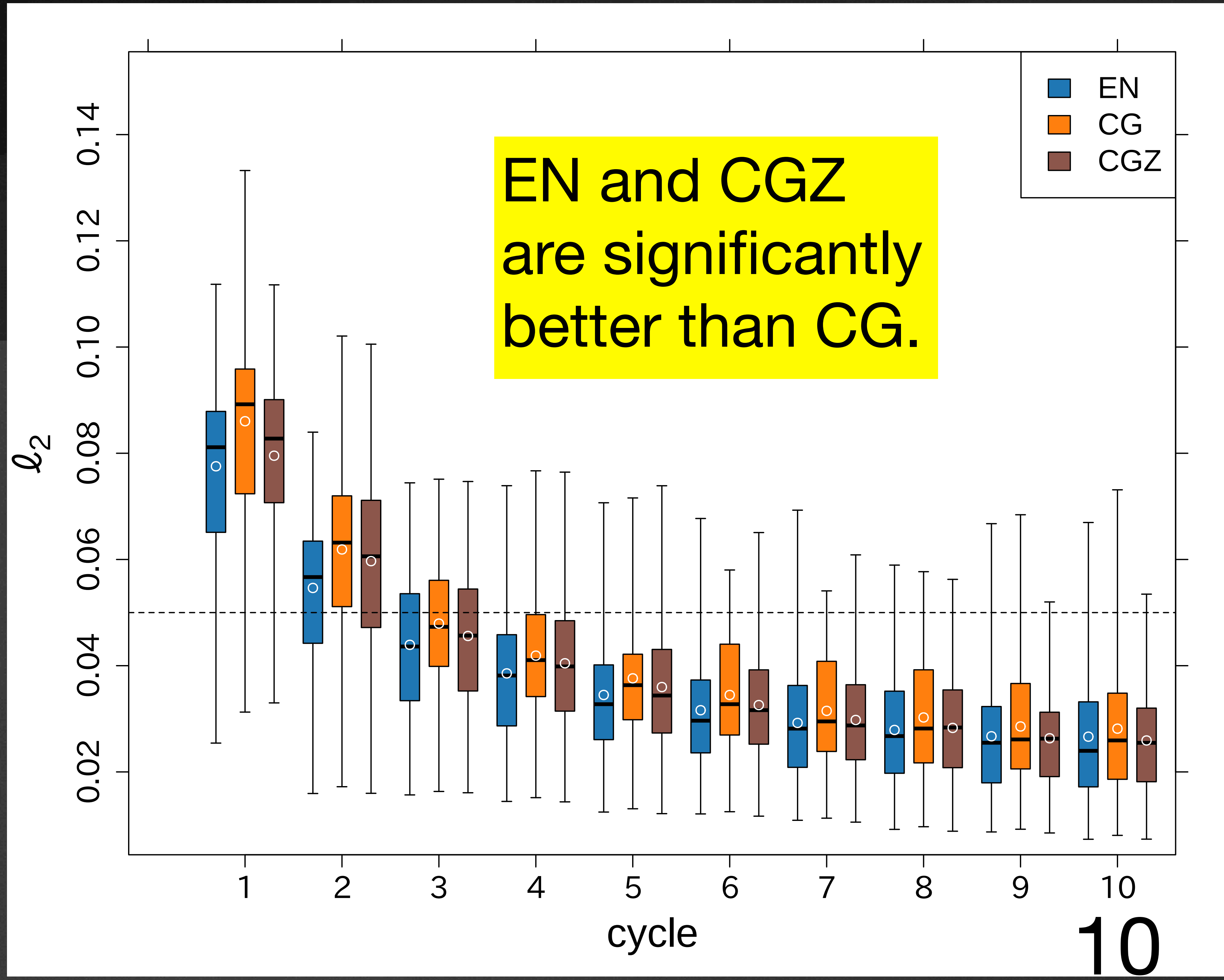
Number of iterations



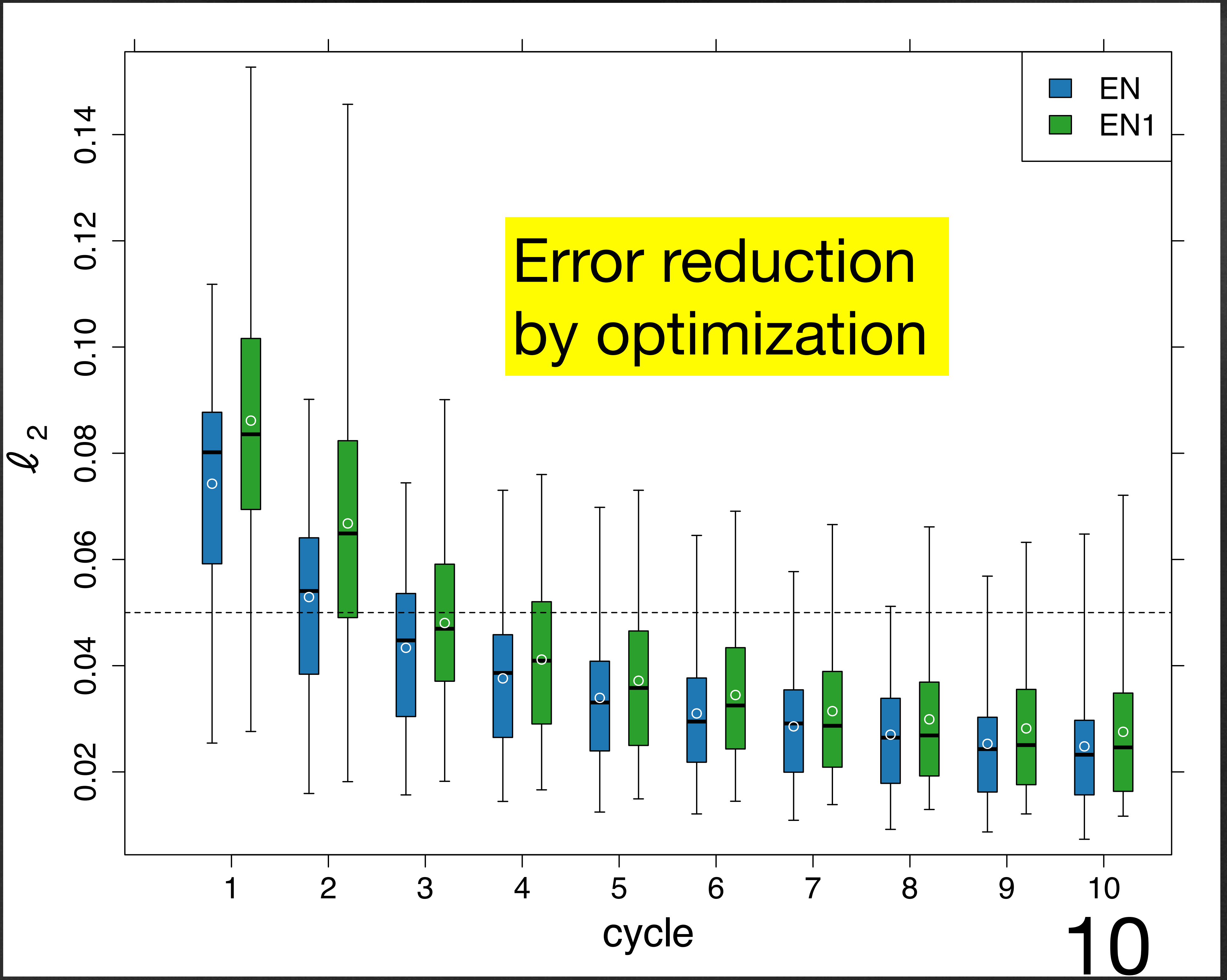
Cycled experiments with a KdVB model

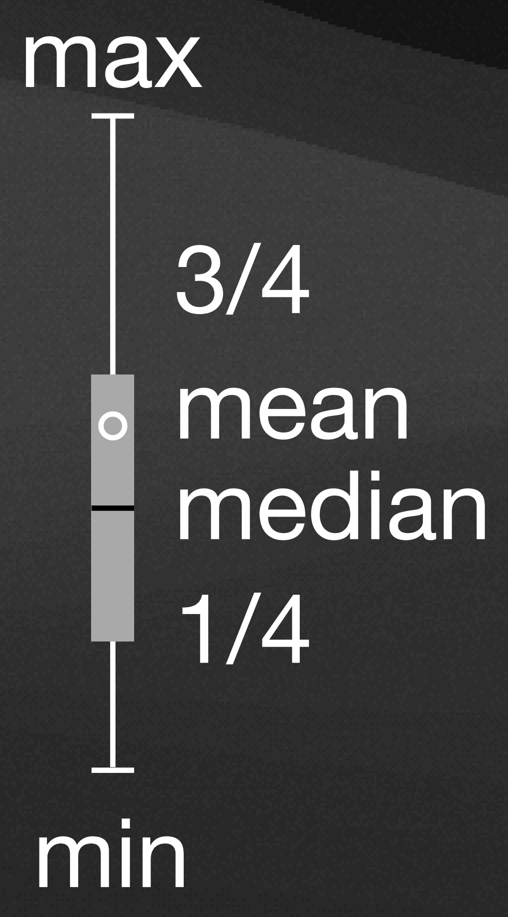
Repeated experiments

EN vs CGZ vs CG (78/100)



EN vs EN1 (44/100)





Summary

Submitted to *Tellus A*

https://github.com/tenomoto/kdvb

- The exact Newton (EN) and conjugate gradient method (CG) methods are compared under the framework of the maximum likelihood ensemble filter (MLEF, Zupanski 2005).
- The Hessian preconditioning works perfectly for the Booth function but not for Rosenbrock function, which can be minimized in five steps with EN.
- In a single wind speed assimilation (Lorenc 2003, Bowler et al. 2013), CG with updated $\mathbf{Z}$ stagnates due to a line search failure.
- EN and CG with updated $\mathbf{Z}$ yield significantly better analysis with a KdVB model and found to be more stable.