Minimization methods in inverse modelling

TM-meeting 1/2-12-2008

Pim Hooghiemstra & Maarten Krol

Outline

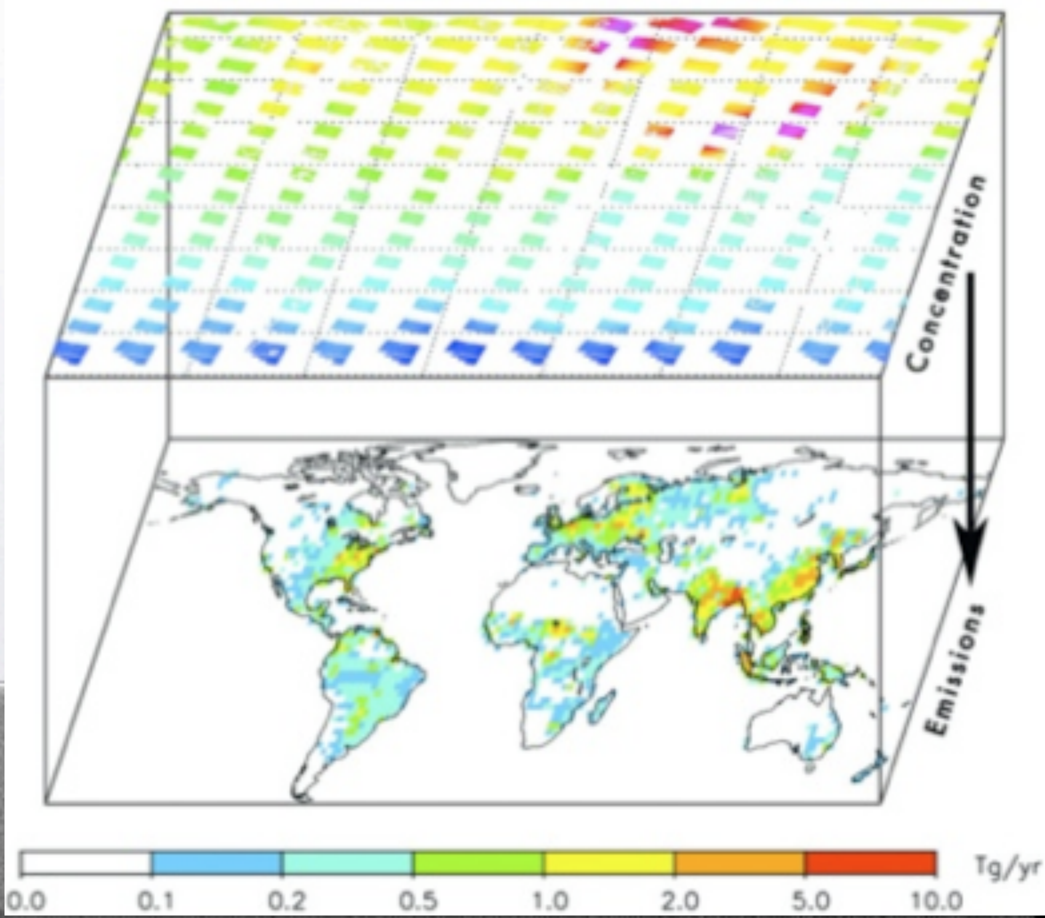
- Inverse modelling
- Linear vs. non-linear inverse modelling
- Minimization methods
- Estimating a posteriori errors

Inverse modelling

Given:

- A set of observations (surface network, aircraft campaigns, satellite instruments) &
- A model to link state vector (emissions) to observations (TM5)

→ Adjust state vector to minimize discrepancy between the observations and model prediction



The Cost function

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - \mathbf{y}_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - \mathbf{y}_i)$$

Regulation term

(This term is added to constrain the problem better.)

Discrepancy between model and observations

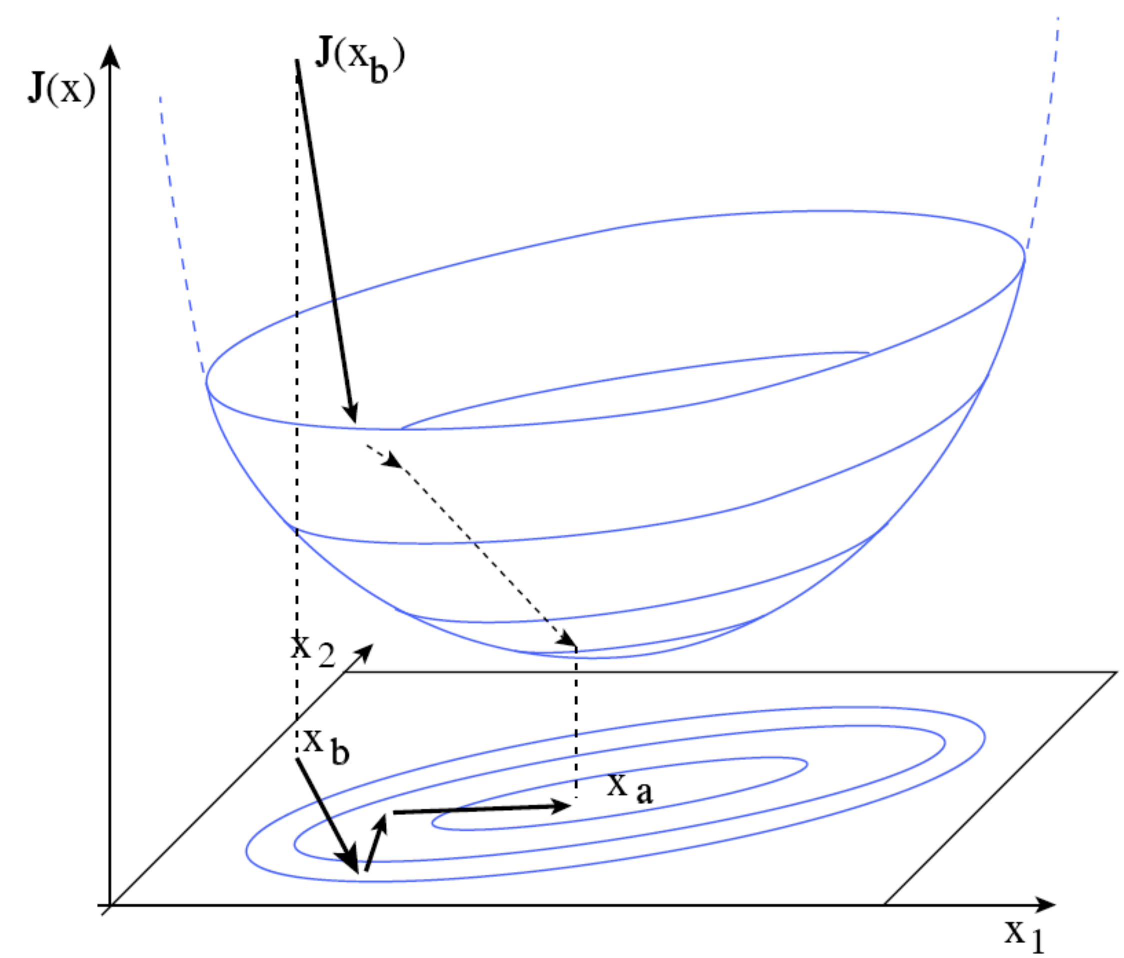
The Cost function

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i-\mathbf{y}_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i-\mathbf{y}_i)$$

If the model H_i is linear, J is a quadratic function of the elements of the state vector x.

Minimize J using the conjugate gradient (CONGRAD) method.

Next: discuss minimization of J for a toy-application



A first inversion

- Build a model to simulate emissions and transport
- Construct observations y by running the model with the initial state.
- Estimate prior by smoothing the initial state.
- Minimize the cost function J with CONGRAD.

Toy model

Model domain consists of 14 grid cells. Cells 0,11-13 do not emit, only transport. The boundary conditions are periodic. The time window is 8 weeks, with a 1 day time step.

What do we optimize?

Initially, no biomass burning (B)

	F	F	F	B	B	B	B	B	F	F			
	0.20	0.35	0.25	0	0	0	0	0	0.40	0.35			
0	1	2	3	4	5	6	7	8	9	10	11	12	13

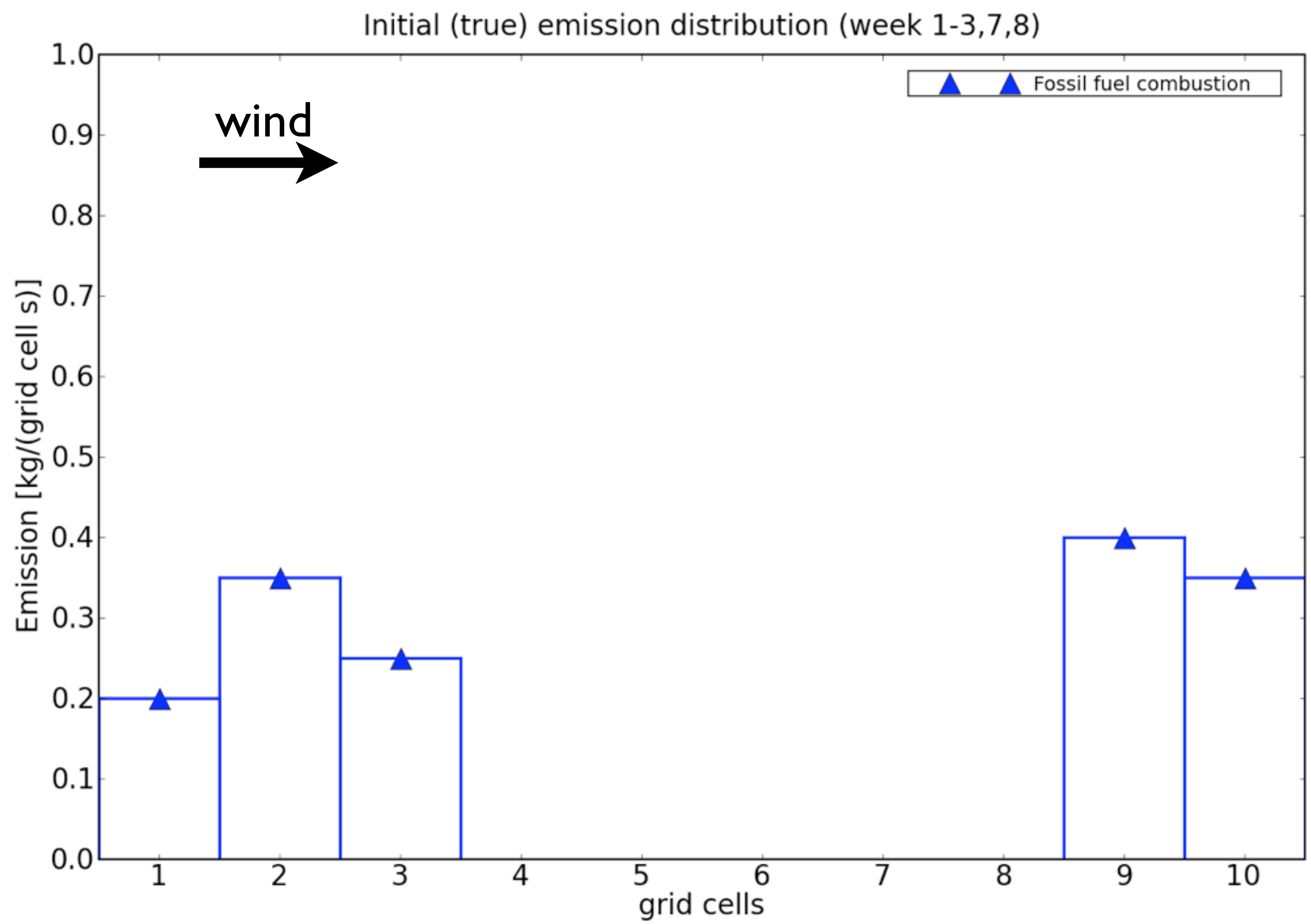
Biomass burning season (week 4-6)

	F	F	F	B	B	B	B	B	F	F			
	0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35			
0	1	2	3	4	5	6	7	8	9	10	11	12	13

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - \mathbf{y}_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - \mathbf{y}_i)$$

Toy model

- 1D tracer model
- transport & emissions
- 2 kinds of emissions variable in time
- wind from left to right

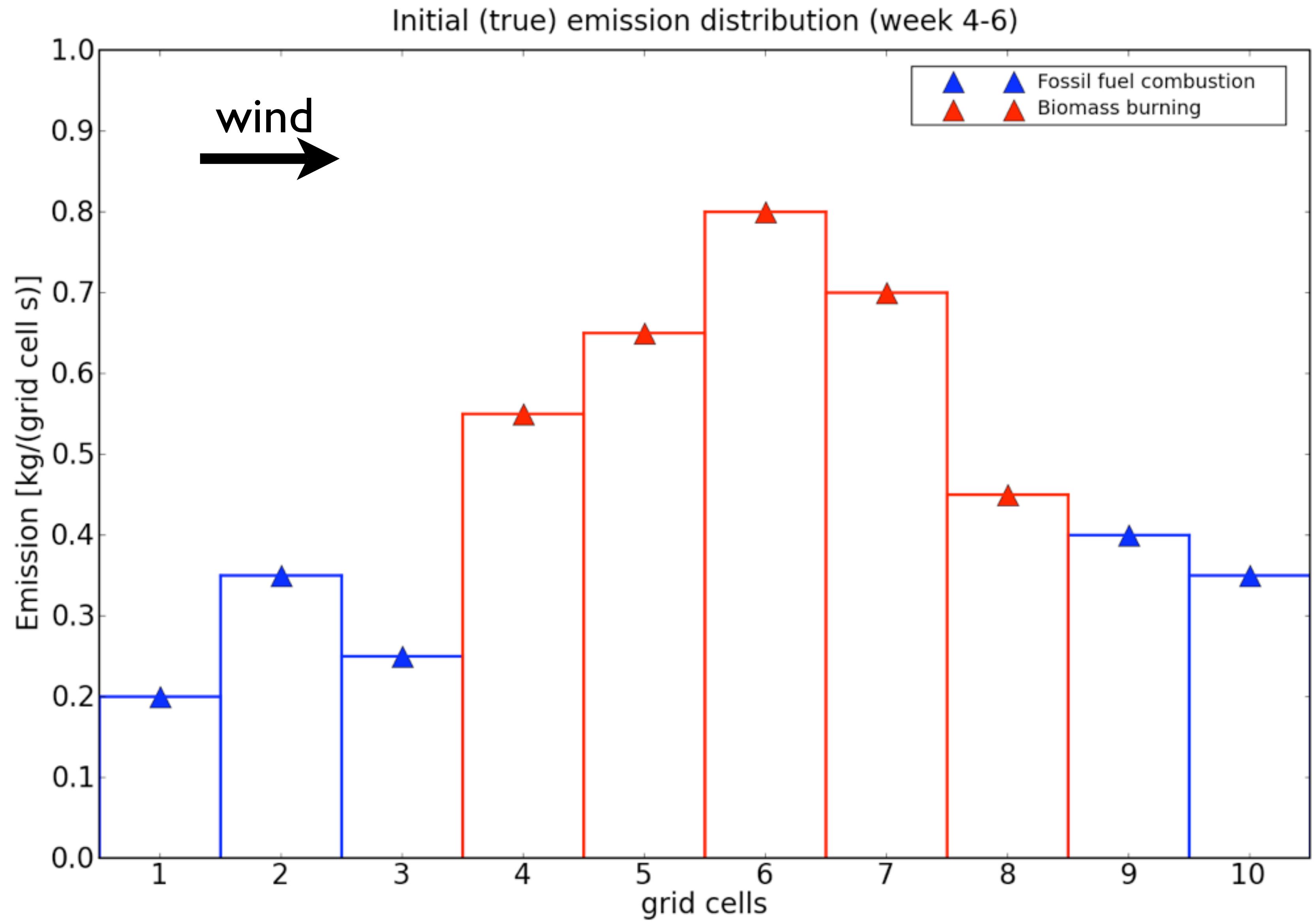


F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0	0	0	0	0	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - \mathbf{y}_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - \mathbf{y}_i)$$

Toy model

- 1D tracer model
- transport & emissions
- 2 kinds of emissions variable in time
- wind from left to right



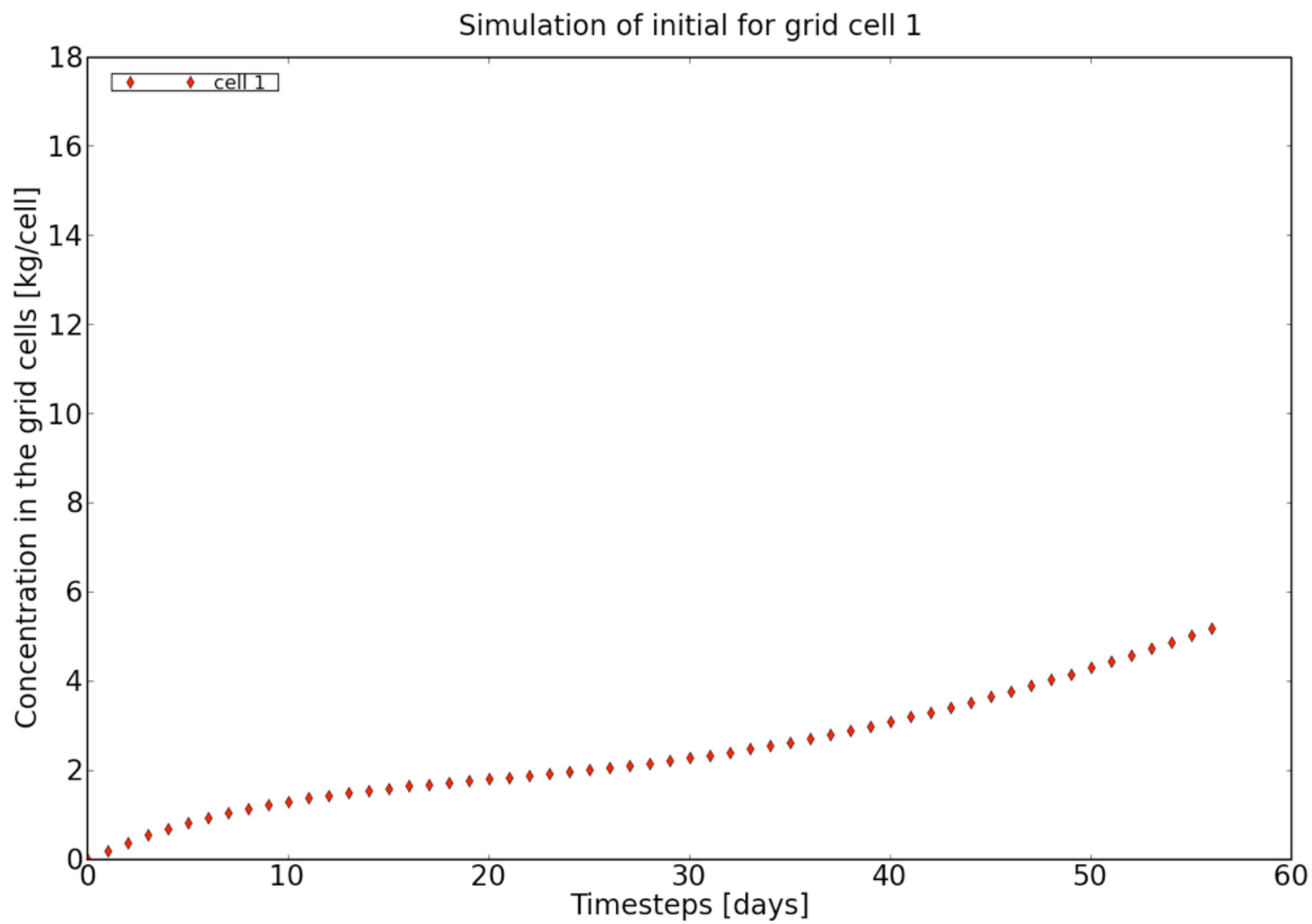
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - y_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - y_i)$$

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10



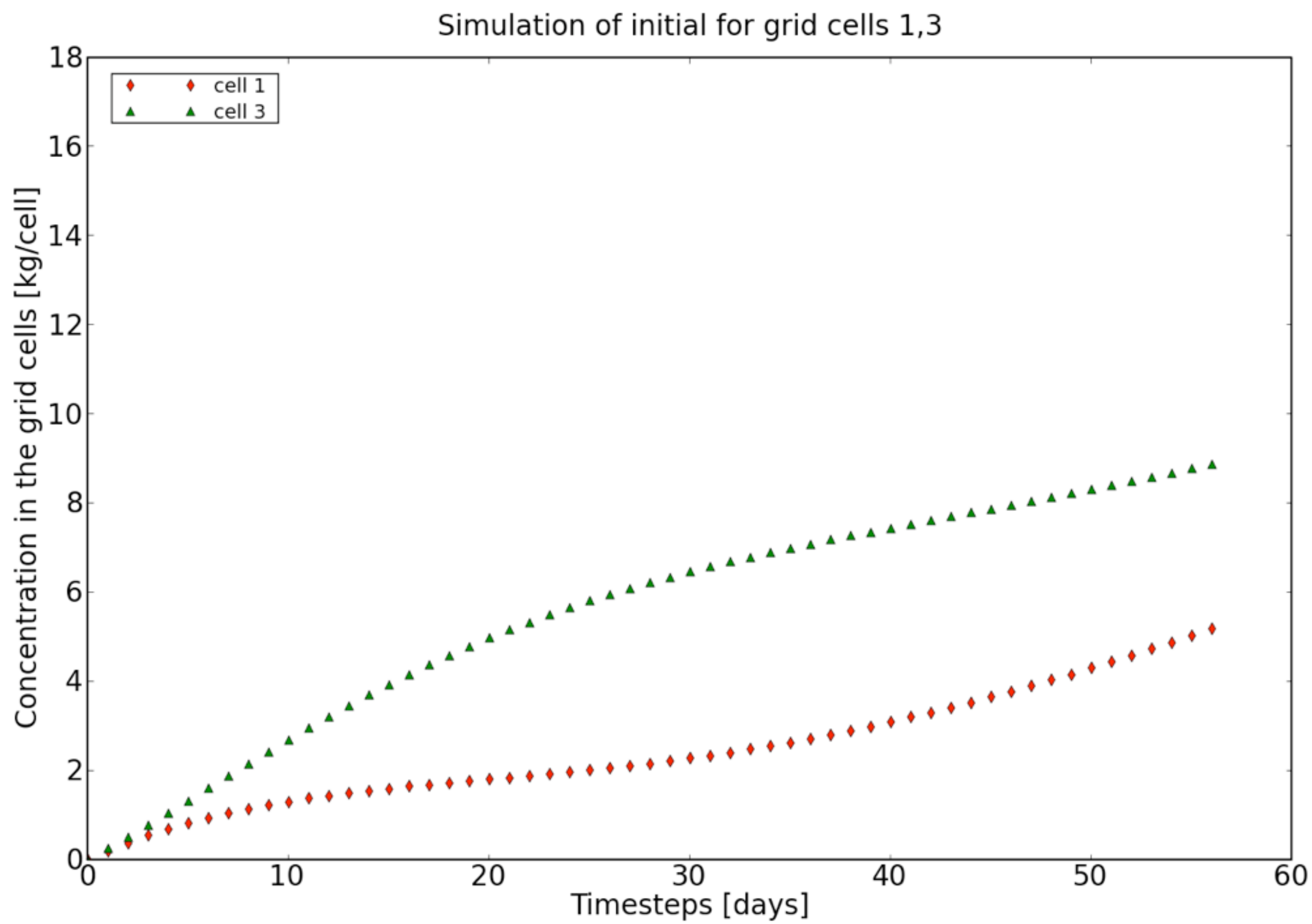
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - y_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - y_i)$$

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10



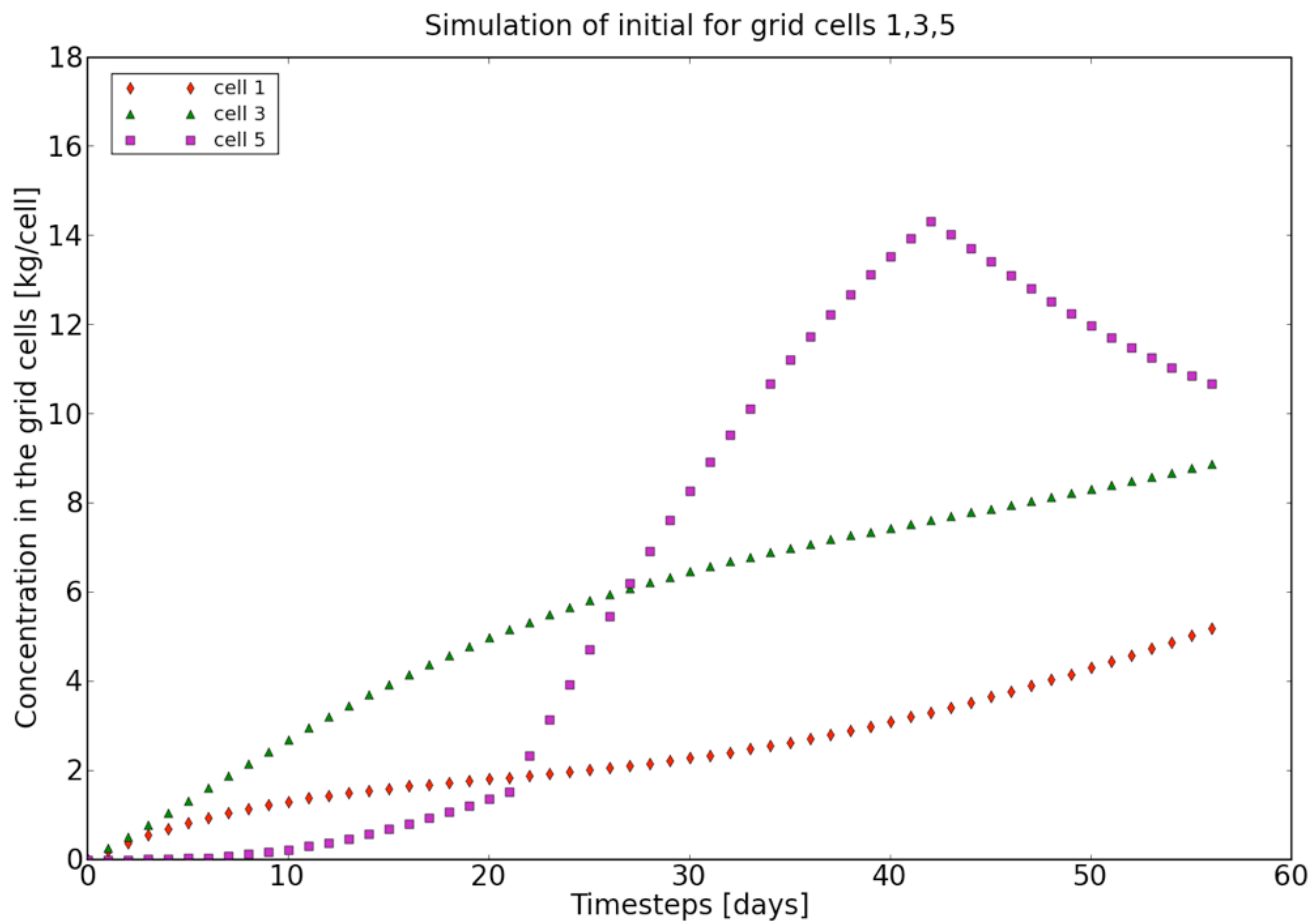
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - y_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - y_i)$$

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10



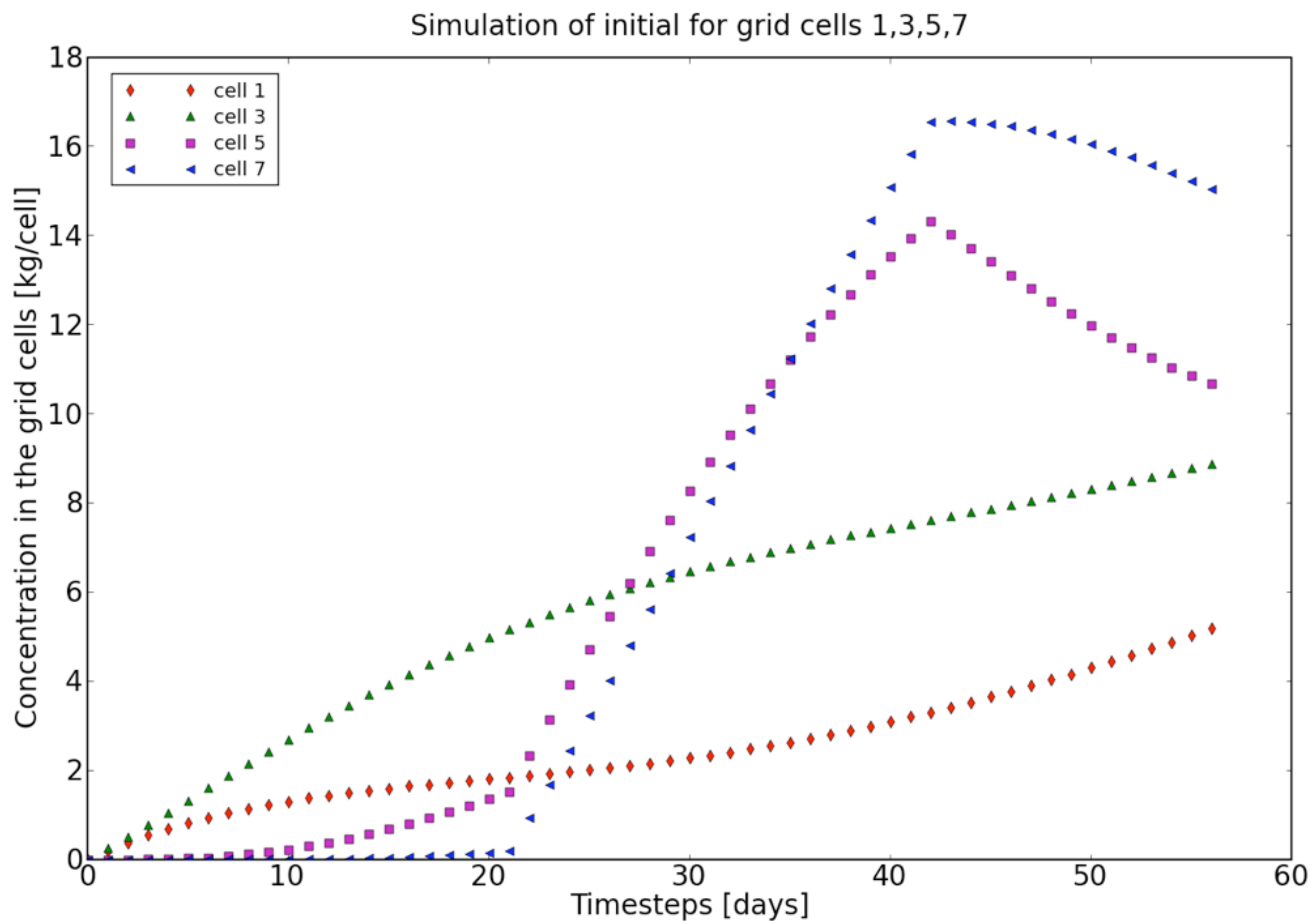
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - y_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - y_i)$$

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10

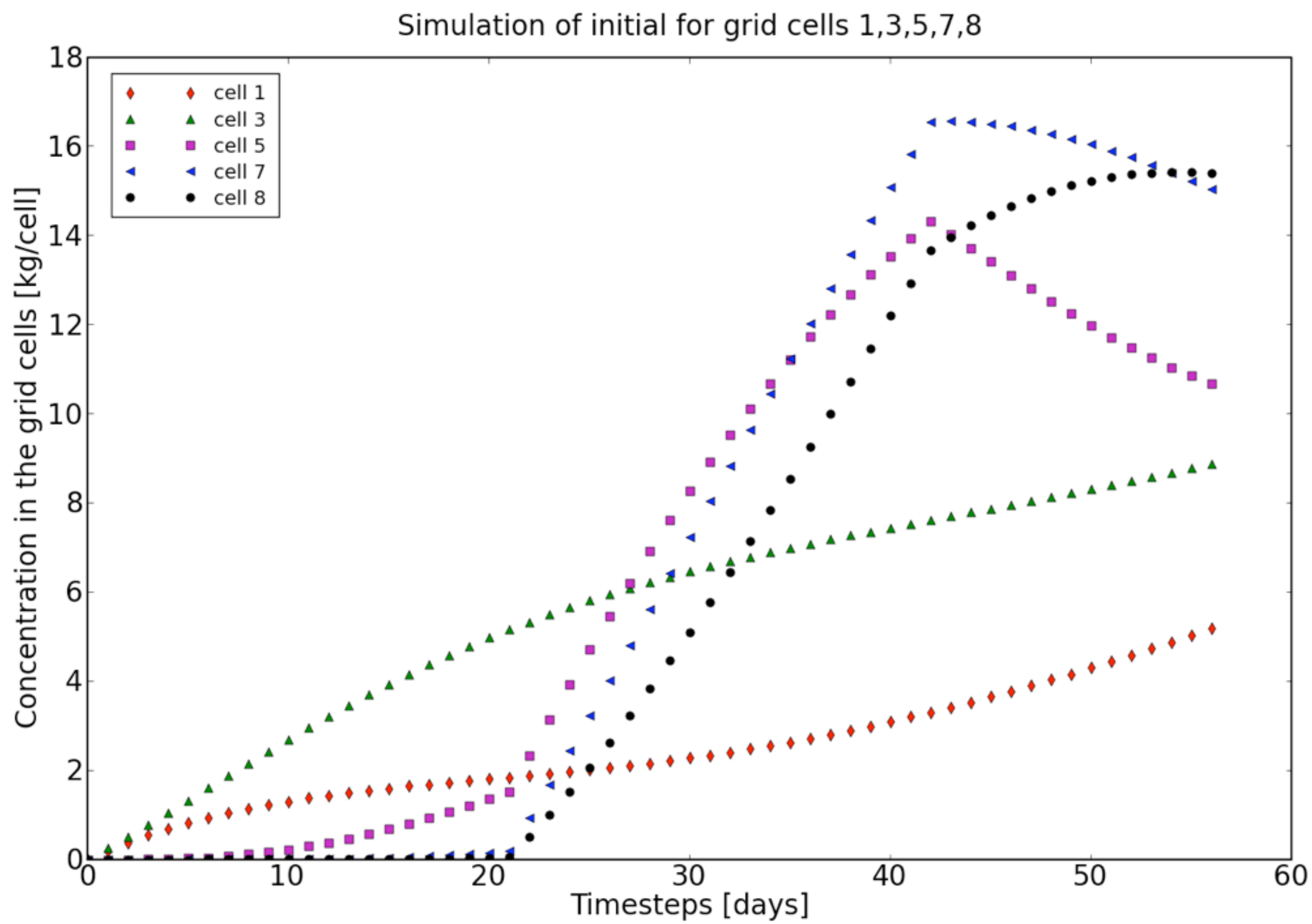


F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10



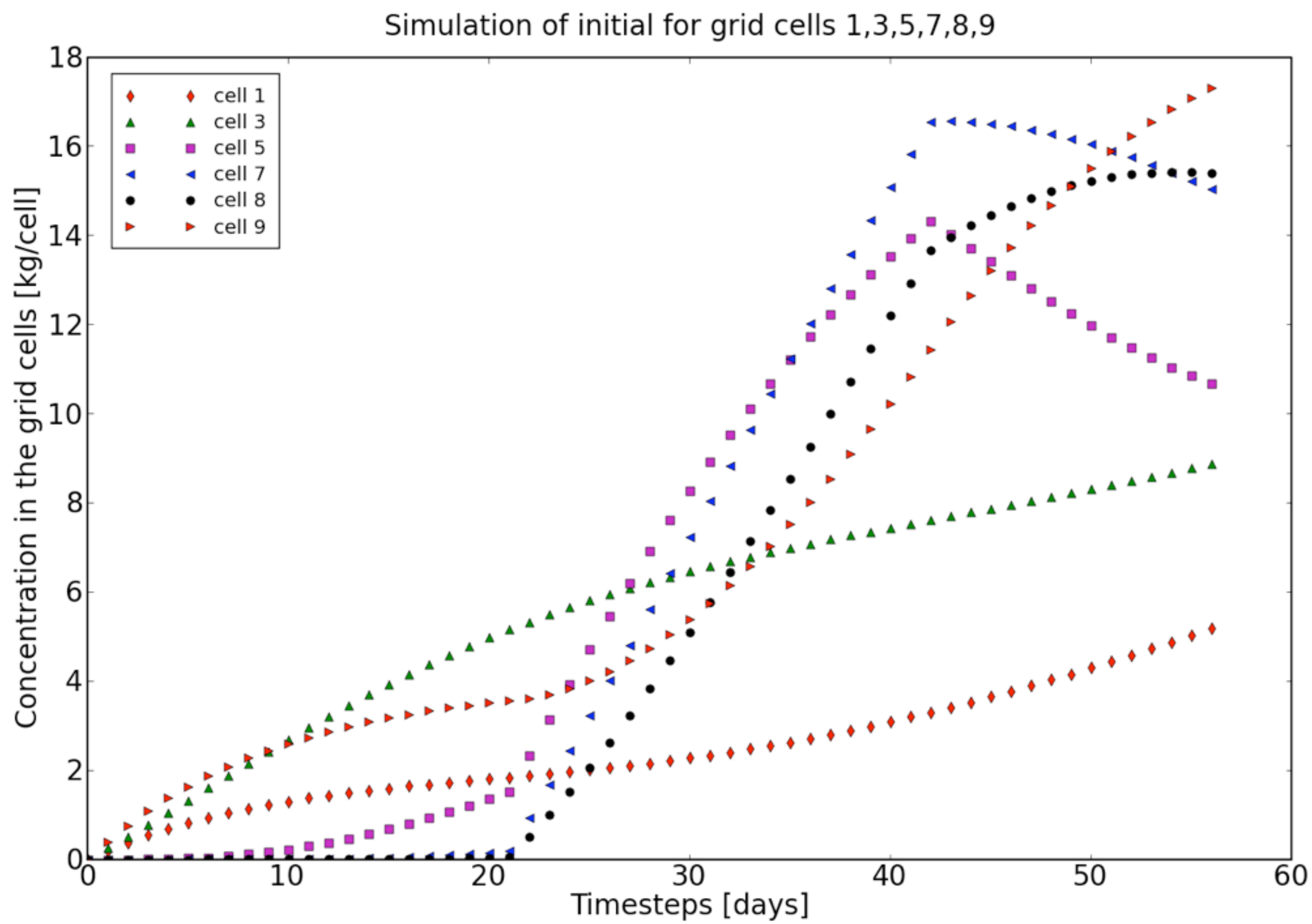
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - y_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - y_i)$$

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10



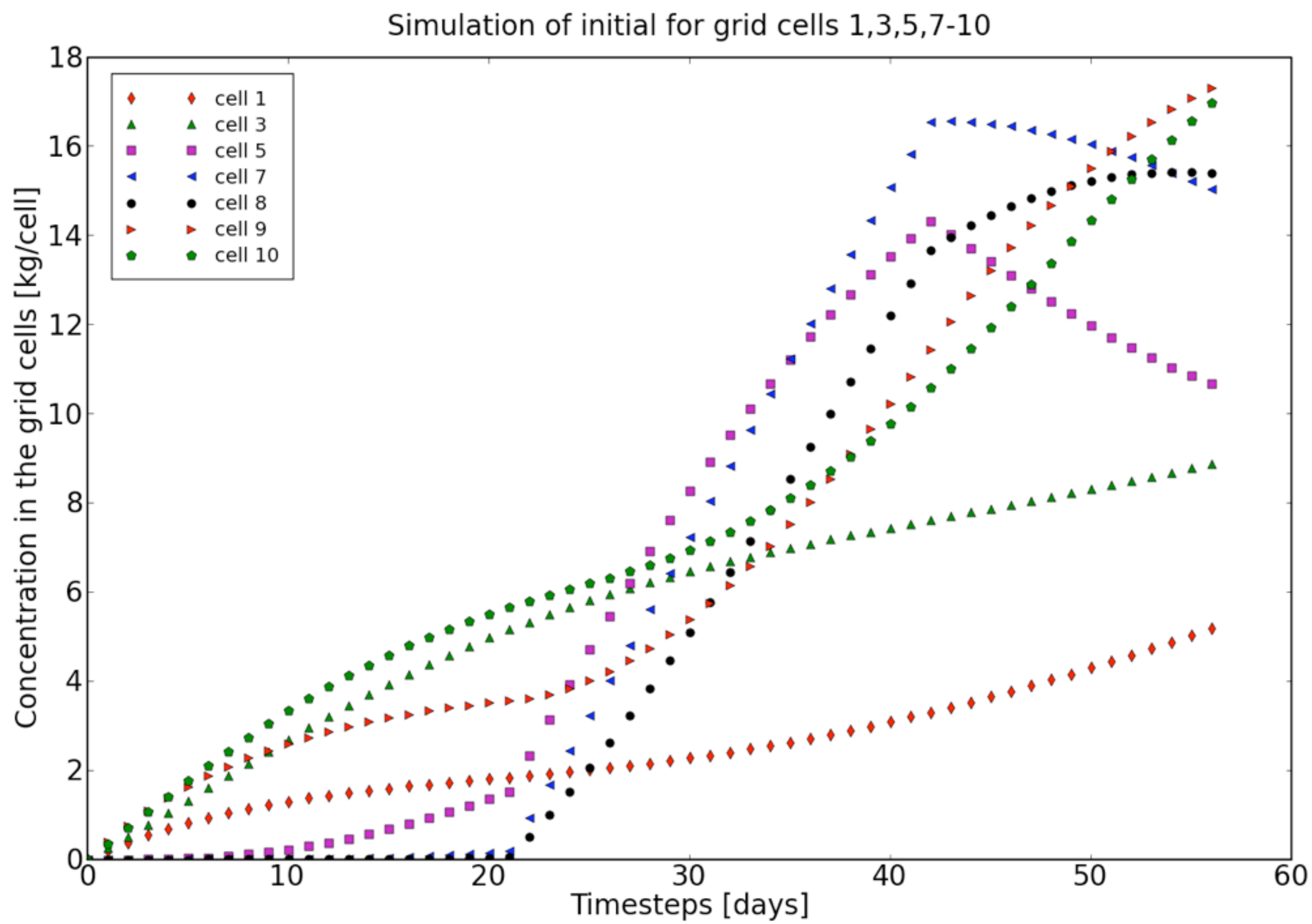
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - y_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - y_i)$$

Model simulation

Run the model with initial state as input to get measurements.

Measurements are taken in grid cells 1,3,5,7-10



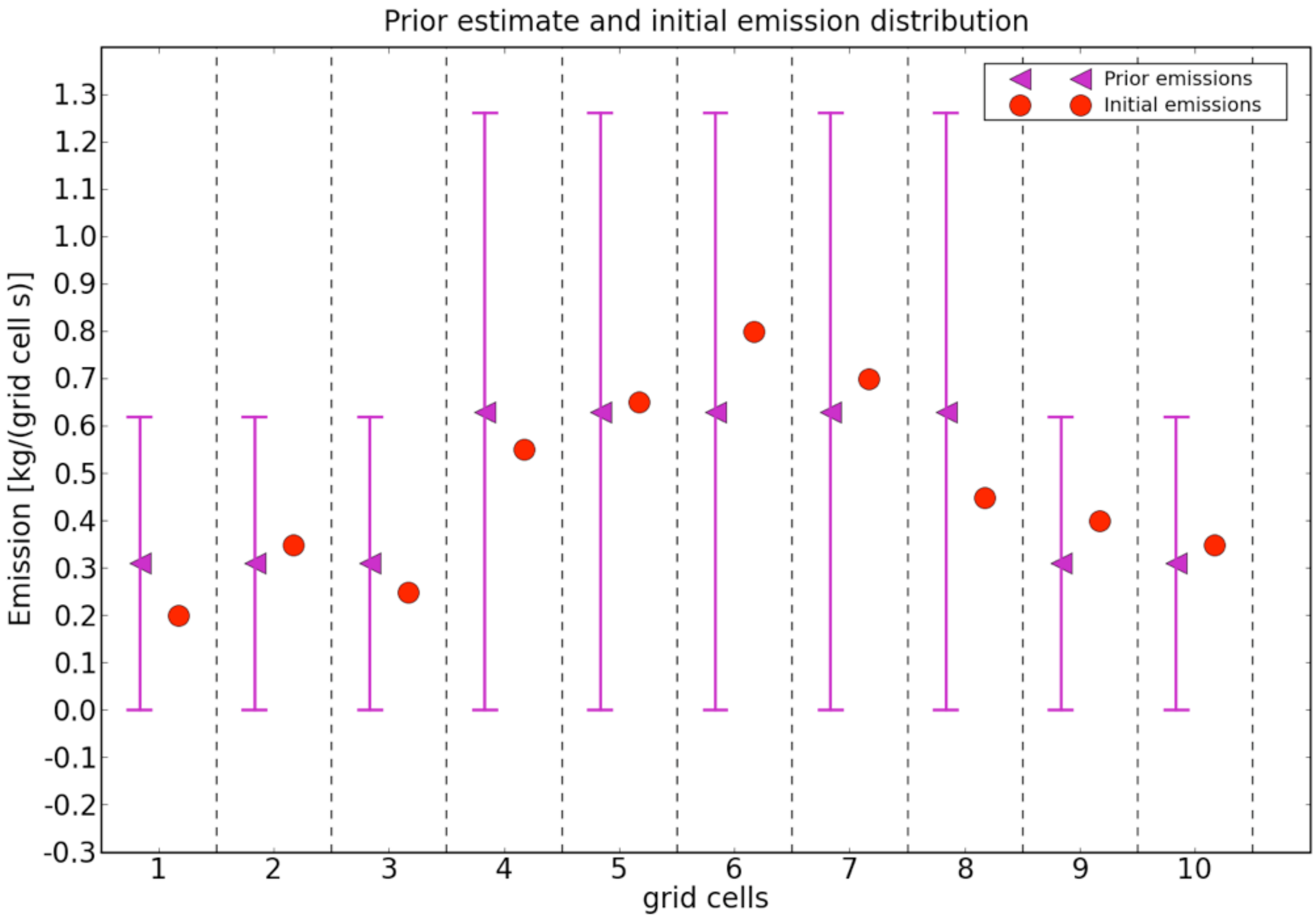
F	F	F	B	B	B	B	B	F	F
0.20	0.35	0.25	0.55	0.65	0.80	0.70	0.45	0.40	0.35
1	2	3	4	5	6	7	8	9	10

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^\top \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}\sum_{i=0}^{N}(H_i\mathbf{x}_i - \mathbf{y}_i)^\top \mathbf{R}_i^{-1}(H_i\mathbf{x}_i - \mathbf{y}_i)$$

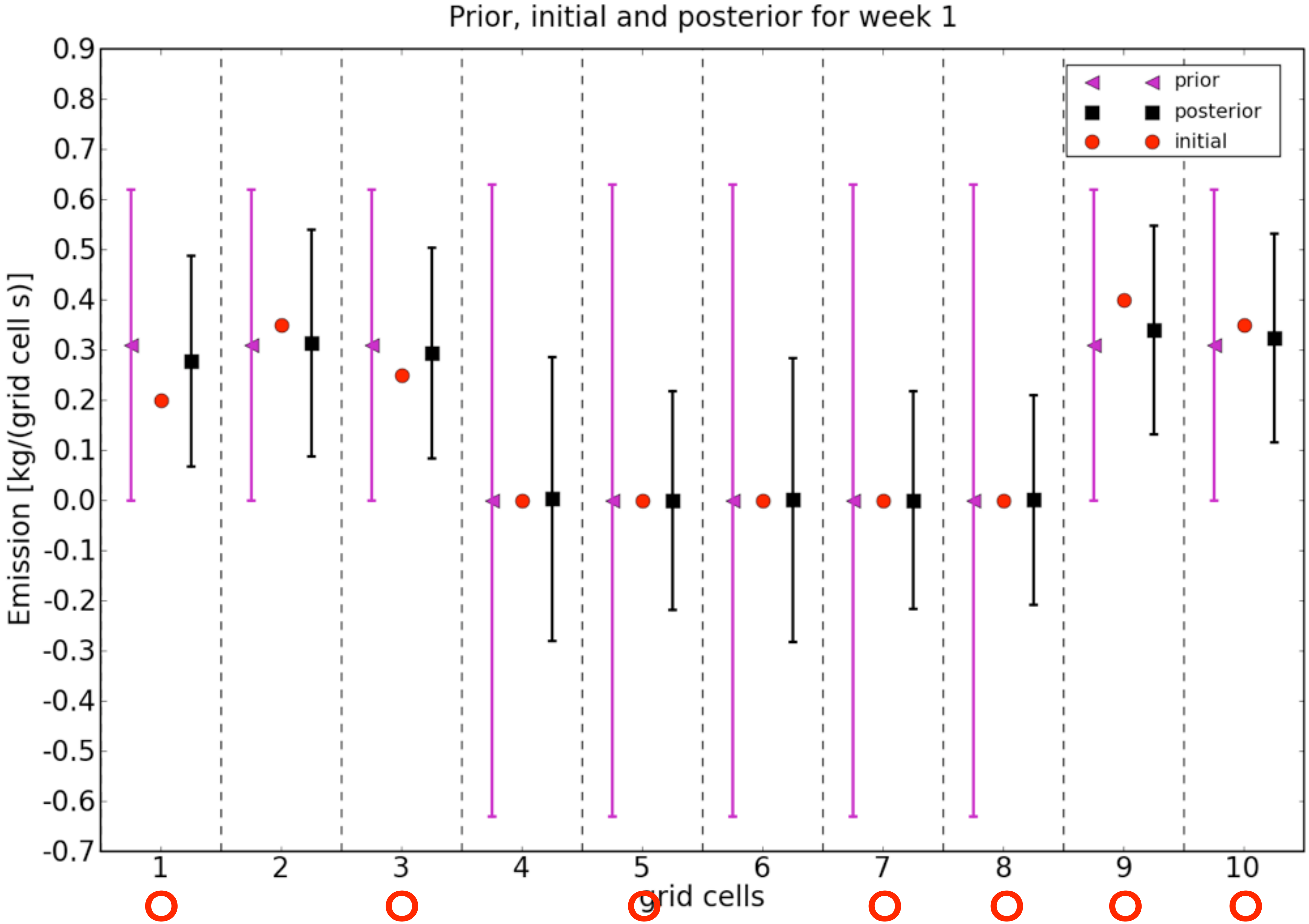
Construction of prior

Smooth initial state per source

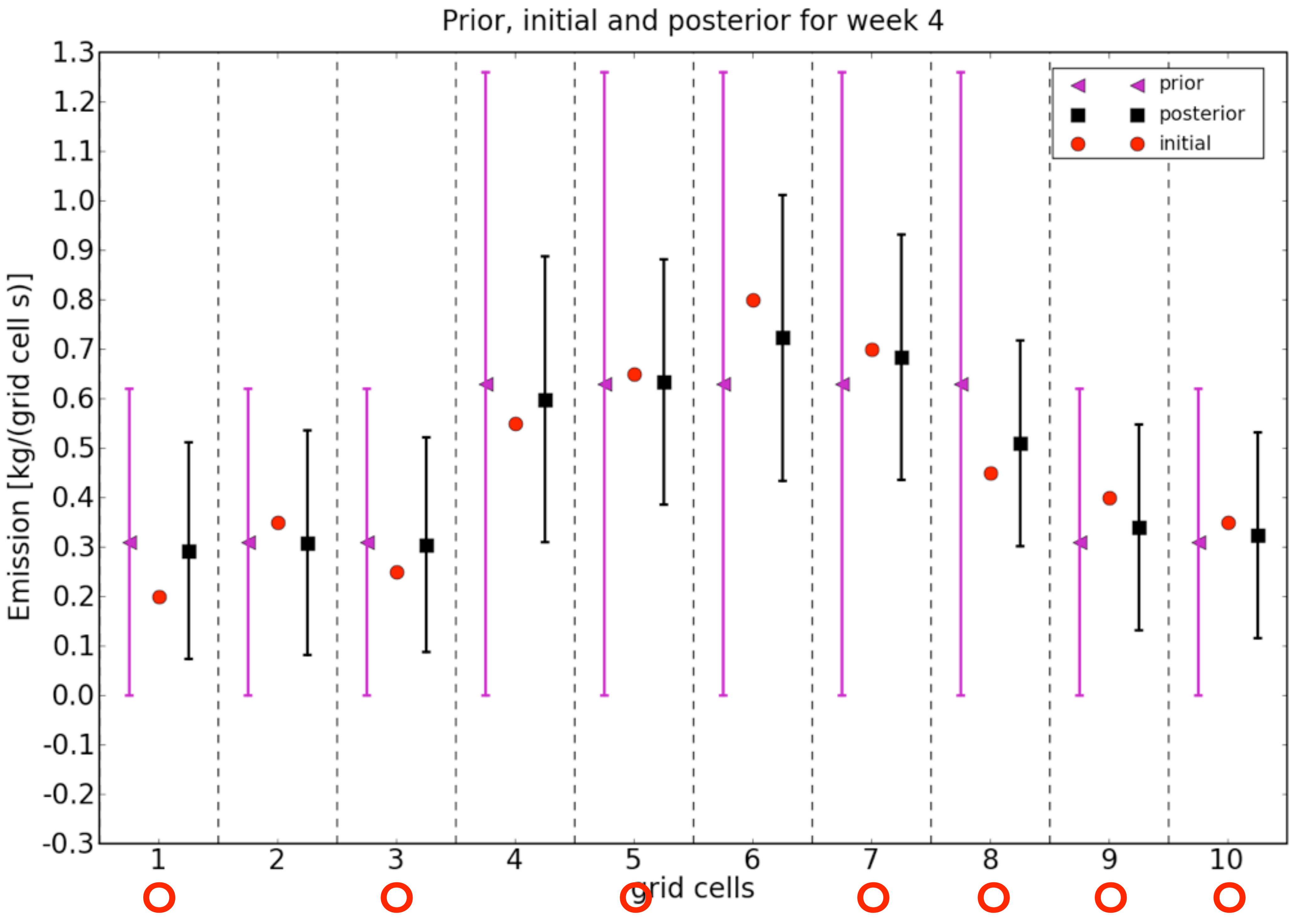
Error on prior FF emissions = 0.31
BB emissions = 0.63



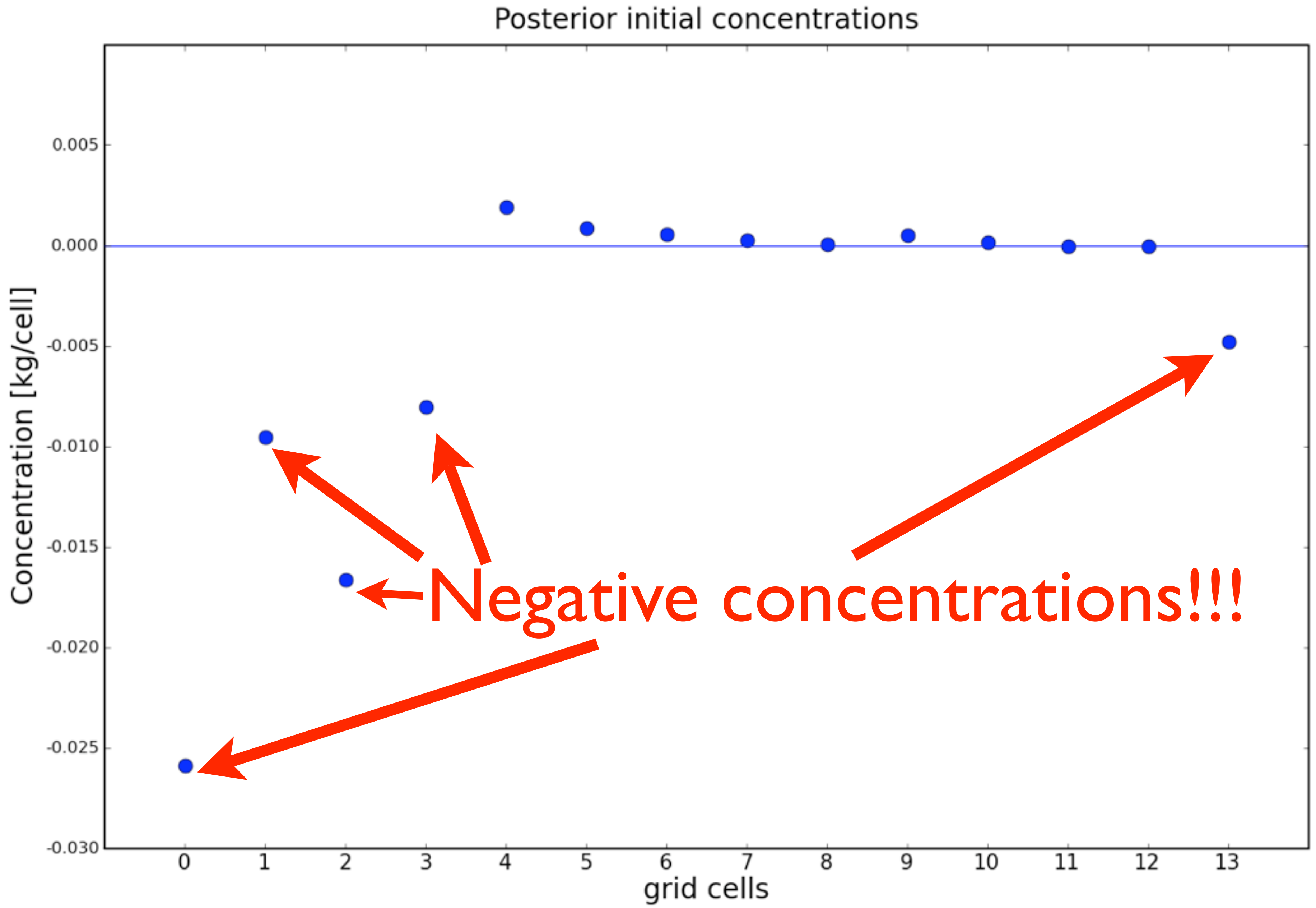
Inversion result

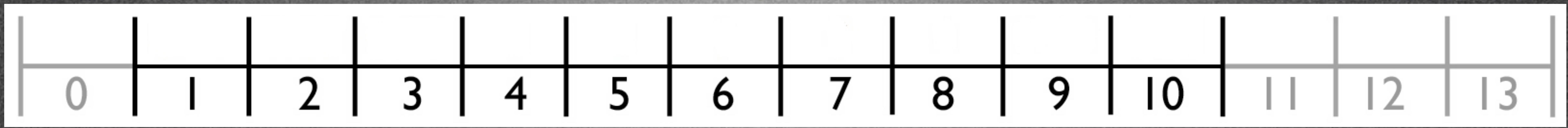


Inversion result



First result: Initial concentration





Solution to negative emissions

- Introduce a set of dimensionless parameters f.
- Let the prior emissions (category 1) be described by

$$\mathbf{x}_b = [C(x), E_1(x,t), E_2(x,t)]$$

- Define posterior emissions by

$$\mathbf{x}_a = \exp(f)\,[C(x), E_1(x,t), E_2(x,t)]\,, \quad f < 0$$
$$\mathbf{x}_a = (1+f)\,[C(x), E_1(x,t), E_2(x,t)]\,, \quad f > 0$$

- and optimize the parameters f, ensuring positive emissions

Solution to negative emissions

Problem: model becomes non-linear and the cost function becomes non-quadratic, the conjugate gradient method can not be used anymore.

We have to use a minimization method capable of solving this kind of problem: M1QN3

Quasi-Newton methods

Method suitable for non quadratic cost function

Developed to minimize functions depending on typically 10^8 parameters.

A posteriori error covariance matrix difficult to estimate.

M1QN3 (version 3.1)

Implementation for the linear case done.
First results: this method converges to the same a posteriori solution as CONGRAD.

Extension to non-linear model not yet implemented.

A posteriori error covariance matrix

After the inversion we get *optimized parameters*, but what about the error in these parameters and their correlations?

Next: Estimating the *posterior error covariance* (*pec*) matrix

From theory: $\mathrm{pec} = \left(\nabla^2 \mathcal{J}\right)^{-1} = \left(\mathbf{B}^{-1} + \mathbf{H}^\top \mathbf{R}^{-1} \mathbf{H}\right)^{-1}$

but it is too CPU consuming to do this computation directly, so we have to approximate the matrix *pec*.

(1) conjugate gradient method

(2) quasi-Newton method

pec matrix CONGRAD

From theory: $\mathrm{pec} = \left(\nabla^2 \mathcal{J}\right)^{-1}$

The conjugate gradient method simultaneously minimizes the cost function J and derives the leading eigenvalues and eigenvectors of the Hessian of J: $\nabla^2 \mathcal{J}$

Hence, we construct an approximation to *pec* using the pairs (λ_i, ν_i)

$\mathfrak{p}$ec matric CONGRAD

From theory: $\text{pec} = \left(\nabla^2 \mathcal{J}\right)^{-1}$

$$\nabla^2 \mathcal{J} = \mathbf{PDP}^\top$$

where,

$$\mathbf{P} = [\nu_1, \nu_2, \ldots, \nu_N]$$

$$\mathbf{D}_{ii} = [\lambda_i]$$

pec matrix M1QN3

Quasi-Newton methods use an approximation to the inverse Hessian as search direction in the algorithm. Now remember:

From theory: $\mathrm{pec} = \left(\nabla^2 \mathcal{J}\right)^{-1}$

So we just construct the matrix used in the M1QN3 algorithm and take it as our approximation for *pec*.

CONGRAD$_{pec}$ vs M1QN3$_{pec}$

For our small problem we can actually compute:

$$\mathrm{pec} = \left(\nabla^2 \mathcal{J}\right)^{-1} = \left(\mathbf{B}^{-1} + \mathbf{H}^{\top}\mathbf{R}^{-1}\mathbf{H}\right)^{-1}$$

Now it turns out that:

(1) CONGRAD$_{pec}$ is exactly equal to THEORY$_{pec}$, but
(2) M1QN3$_{pec}$ is a little bit different.

Matrix aggregation

Aggregation is averaging of the matrix over the spatial or temporal domain.

(1) Convergence of aggregated *pec* matrix faster.
(2) Better comparison for aggregated *pec's* of CONGRAD and M1QN3.

Matrix aggregation

(1) Convergence of aggregated *pec* matrix faster.

Unfortunately, this is not tested yet
work in progress :-(

Matrix aggregation

(2) Better comparison for aggregated *pec's* of CONGRAD and M1QN3.

Left: Initial concentration vs. total emissions. Right: Initial concentration vs. FF emissions vs. BB emissions.

M1QN3

1.00000	-0.80475
-0.80475	1.00000

1.00000	-0.16352	-0.43586
-0.16352	1.00000	-0.72191
-0.43586	-0.72191	1.00000

CONGRAD

1.00000	-0.49959
-0.49959	1.00000

1.00000	-0.06463	-0.12803
-0.06463	1.00000	-0.92534
-0.12803	-0.92534	1.00000

Conclusion

- Iterative minimization methods needed as direct computation is often infeasible.

- CONGRAD works fine for linear models: Convergence of *pec* matrix is slow, but we can aggregate.

- M1QN3 not yet tested for a non-linear model and aggregation of the *pec* matrix does not compare to CONGRAD.