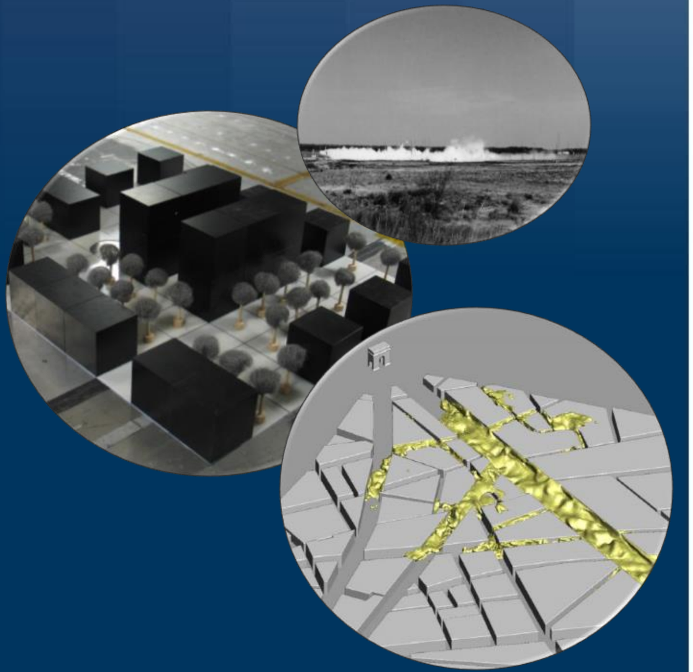
MODITIC

Modelling the dispersion of toxic industrial chemicals in urban environments

INVERSE MODELLING IN URBAN ENVIRONMENTS



Harmo'17

12.05.2016

Niklas Brännström, Tobias Brännvall, Stéphane Burkhart, Jan Burman, Xavier Busch, Jean-Pierre Issartel and Leif Å. Persson

FFI Forsvarets forskningsinstitutt
Norwegian Defence Research Establishment

FOI

DGA

INERIS

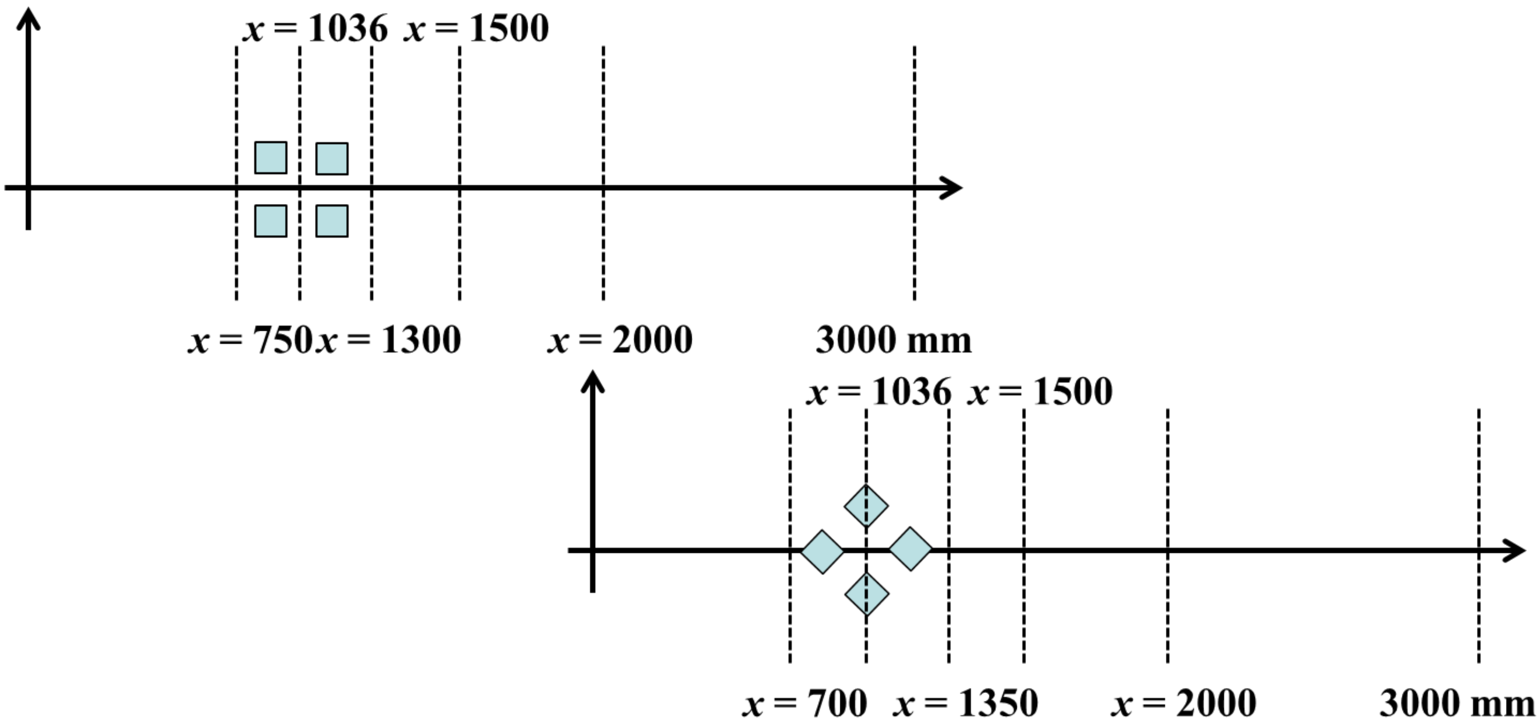
UNIVERSITY OF SURREY

Contents

- Experimental setup
- Source-sensor relationship
- Model and sensor data, distance functions
- The renormalization method
- The normalized least squares method
- The Bayesian view on least squares
- Results, model comparisons
- Conclusions and remarks

FOI

Measurement stations, 4 FFIDs



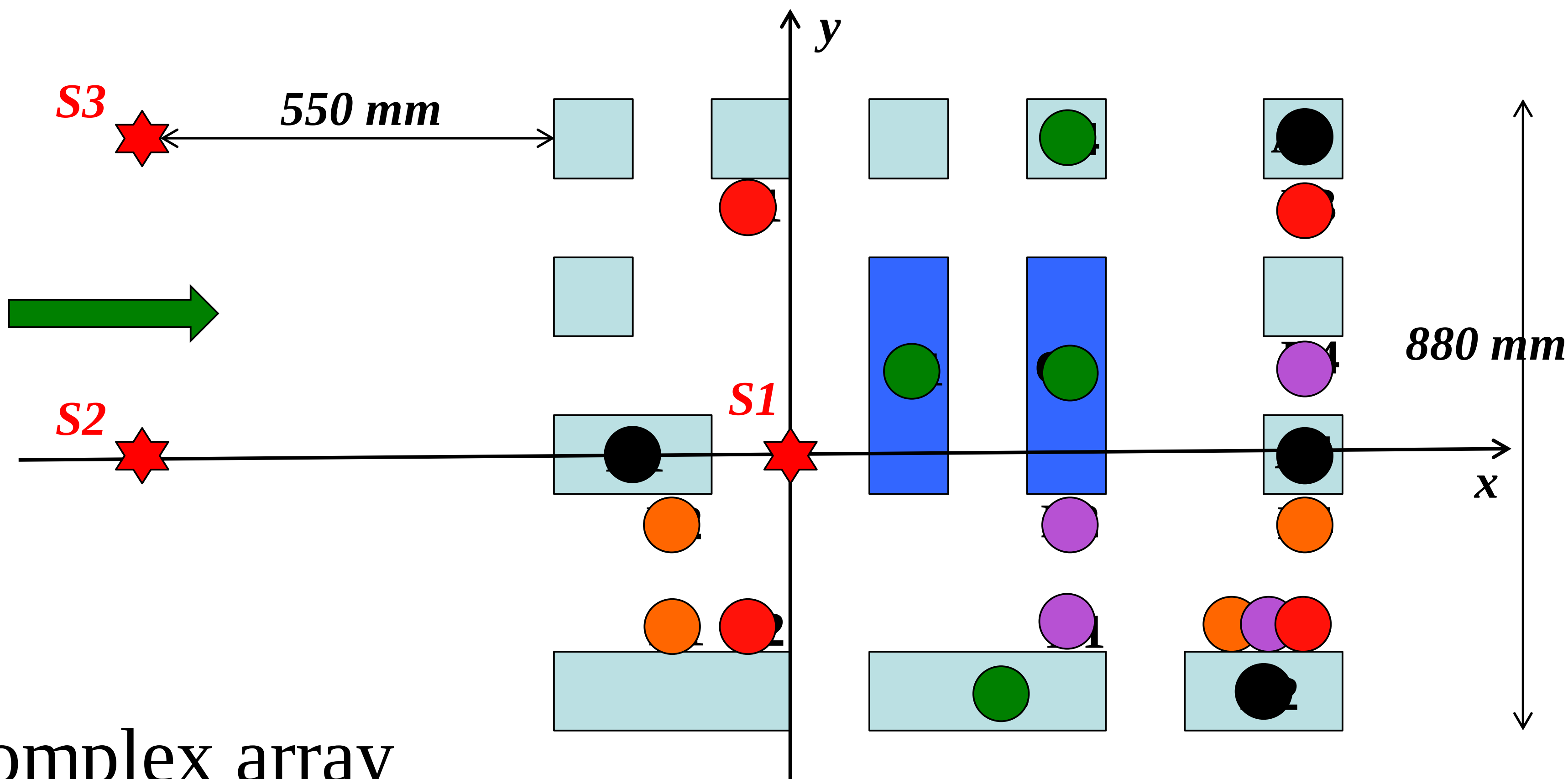
Array A: *(1400, 0), (1600, 0), (2000, 0), (3000, 0)*
Array B: *(2000, -320), (2000, -100), (2000, 100), (2000, 320)*
Array C: *(2000, -320), (2000, 320), (3000, -320), (3000, 320)*
Array D: *as A with rough surface*

Simple array

Arrays used for 4FFID experiments

Detector arrays

- ***Array A (roof level)***
- ***B (street level)***
- ***C (roof)***
- ***D (street)***
- ***E (street)***



Complex array

Source-sensor relationship

- Stationary plume from continuous point source with constant release rate
- Source parameters to estimate: *x,y,z,q*
- Forward model plume: fixed source, sensor positions at grid points
- Adjoint model plume: fixed sensor, source positions at grid points
- Adoint model usually preferred for source estimation for computational efficiency

FOI

Model and measured data

- Sensors numbered $i=1,2,3,4$
- Model data: $c_i = q\, \chi_i(x,y,z)$
- $\chi_i(x,y,z)$ grid function, adjoint plume for sensor i
- Model data vector $\mathbf{c} = [c_1, c_2, c_3, c_4]$
- Measured data vector $\mathbf{d} = [d_1, d_2, d_3, d_4]$
- Adjoint plume vector function
  $\boldsymbol{\chi} = [\chi_1(x,y,z), \chi_2(x,y,z), \chi_3(x,y,z), \chi_4(x,y,z)]$

FOI

The source estimation problem

- Optimization problem
- Find *(x,y,z,q)* minimizing *distance* between model data vector ***c****(x,y,z,q)* and measured data vector ***d***
- Methods differ by
  - how distance is defined
  - numerical method for approximating *(x,y,z,q)*

FOI

Distance functions

- Issartel's renormalization distance: a weighted Euclidean norm based on the *visibility function* φ:

$$(\mathbf{d} - \mathbf{c})^T H_\varphi^{-1} (\mathbf{d} - \mathbf{c})$$

- Normalized Euclidean distance (Mahalanobis distance) D: $D^2 = \Sigma_\mathrm{i} (d_i - c_i)^2 / c_i^2$

FOI

Visibility function and weights

- The *visibility function* $\varphi(x,y,z)$ is constructed from the adjoint plumes χ_i for all sensors by an entropy minimization principle
- Measuring the ”visibility” from the sensor network
- The elements of the weight matrix H_φ are computed by numerical integration of products $\chi_i \chi_j / \varphi$.

FOI

Computational procedure

- By the particular properties of *φ*, the problem is reduced to a simpler maximization problem:
- Compute *distributed* renormalization source:

$$\sigma = \mathbf{d}^T H_\varphi^{-1} \boldsymbol{\chi}$$

  on grid.
- Compute point source location *(x,y,z)*:

$$\sigma(x,y,z) = max\ \sigma$$

- Compute point source release rate:

$$q = \sigma(x,y,z)/\varphi(x,y,z)$$

Normalized least squares

- Bilevel optimization
- Compute optimal $q = q^*(x,y,z)$ for each (x,y,z) in grid
- Obtain *envelope grid function*

$$c_i^* = q^*(x,y,z)\, \chi_i(x,y,z)$$

- Minimize the *minimum value grid function*

$$V(x,y,z) = \Sigma_i\, (d_i - c_i^*)^2/(c_i^*)^2$$

Normalized least squares with regularization

- Add penalization term $\lambda\ q^*(x,y,z)$, penalizing large release rates:
- Minimize grid function

$$V(x,y,z) = \Sigma_i\ (d_i - c_i^*)^2/(c_i^*)^2 + \lambda\ q^*(x,y,z)$$

- We have used $\lambda=1$ in this study.

FOI

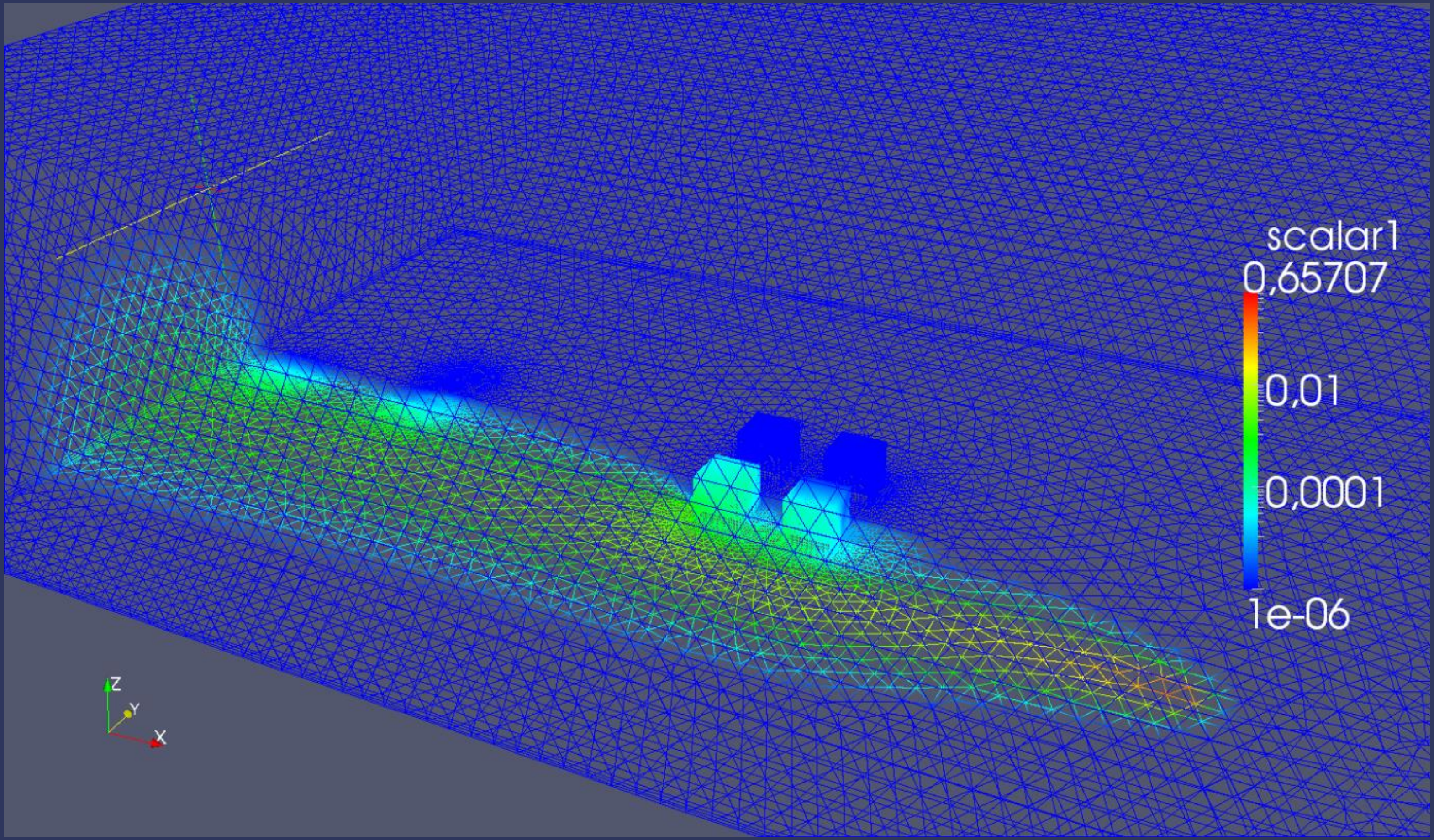
The Bayesian view on LS

- Minimization problem: *min* D^2
- Equivalently, *max* $exp(-D^2)$, may be interpreted as a maximum likelihood estimation (MLE) problem
- By Bayes formula, $exp(-D^2)$ may be interpreted as a posterior density (with noninformative prior)
- Regularization may be introduced as a (non-constant) prior density.

Adjoint plumes

- RANS advection-diffusion models
- Adjoint obtained from forward model by reversing advection and preserving diffusion (self-adjoint)
- RANS-solvers used:
  - Phoenics
  - Code Saturne v4

Adjoint plumes – Code Saturne



Results

- Source estimates *(x,y,z,q)* obtained by maximizing
  - The posterior densities (with or without regularization)
  - The distributed renormalization source
- Isosurface plots of objective functions
- Selected windtunnel data:
  - 5 configurations of the complex array.
  - 6 configurations of the simple array
  - 2-4 datasets/configuration, total number 24
  - PHOENICS adjoints; Code Saturne adjoints pending

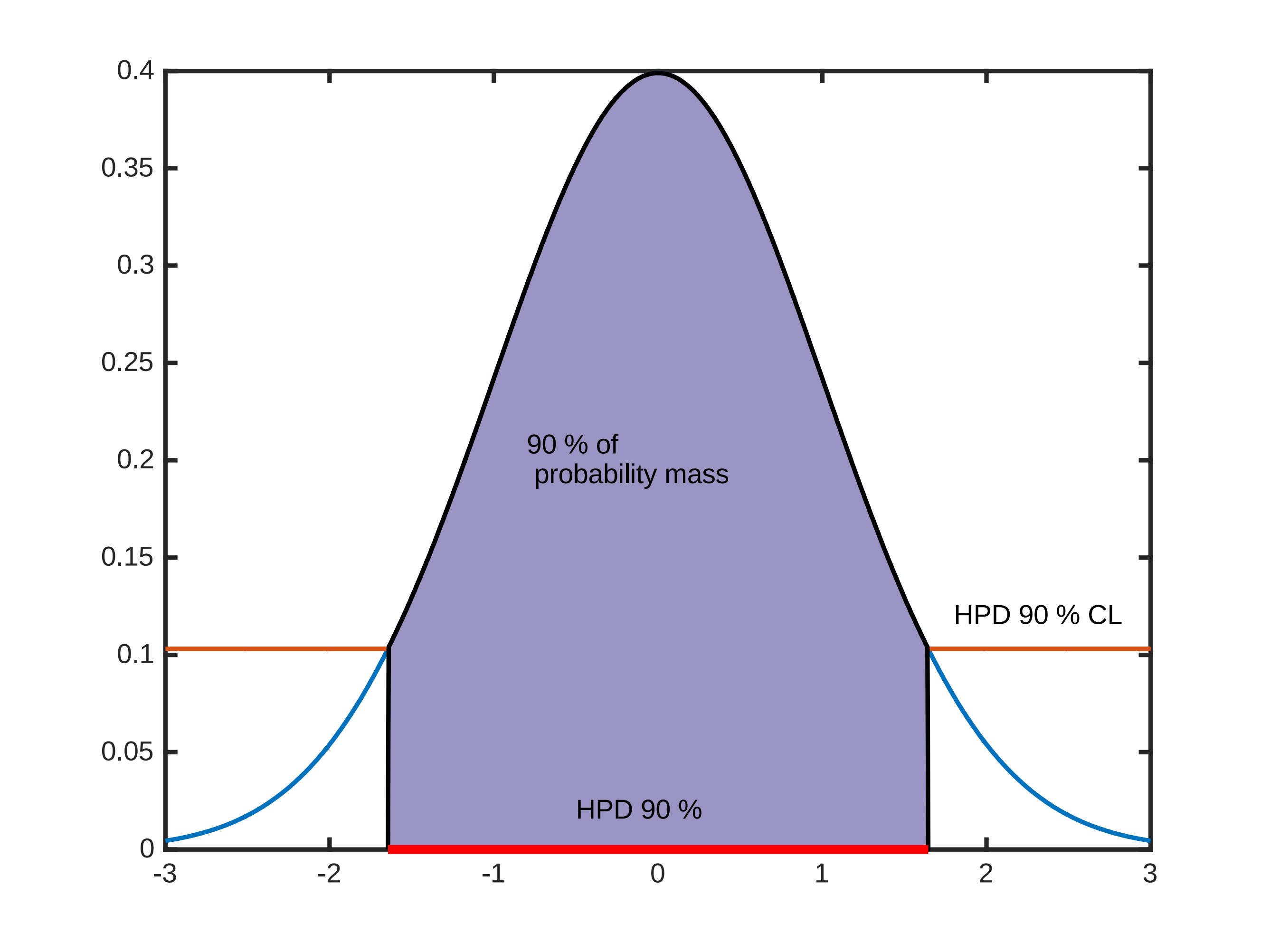
FOI

3D and 2D results

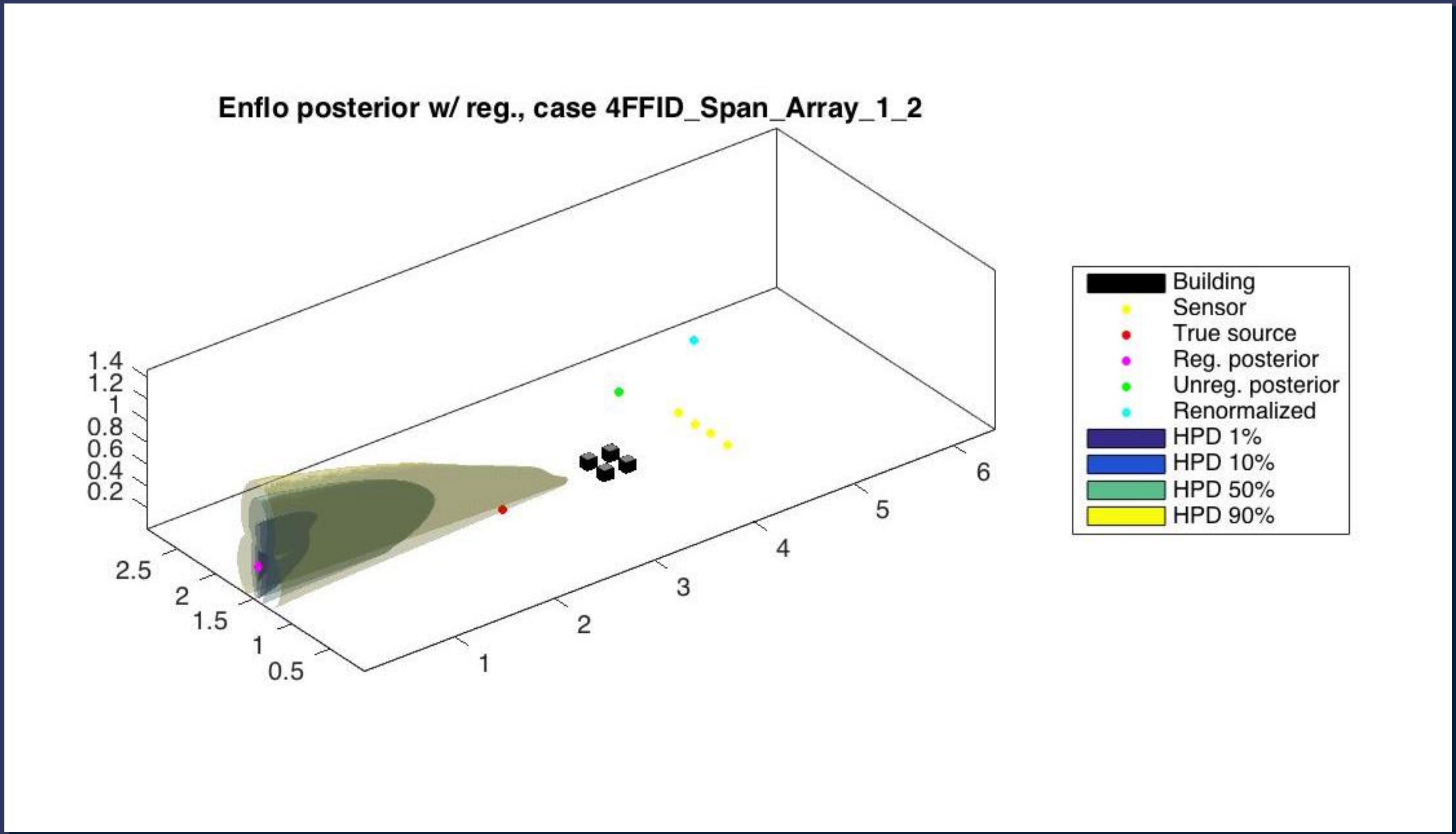
- 3D: Possibly elevated source
  - Source parameters *(x,y,z,q)*
  - Posteriors and renormalization source computed on 3D grid *(x,y,z)*
- 2D: Assuming ground source
  - Source *parameters (x,y,q)*
  - Posteriors and renormalization source computed on 2D grid *(x,y)*, *z=0*
- Level sets presented as highest posterior density (HPD) domains

FOI

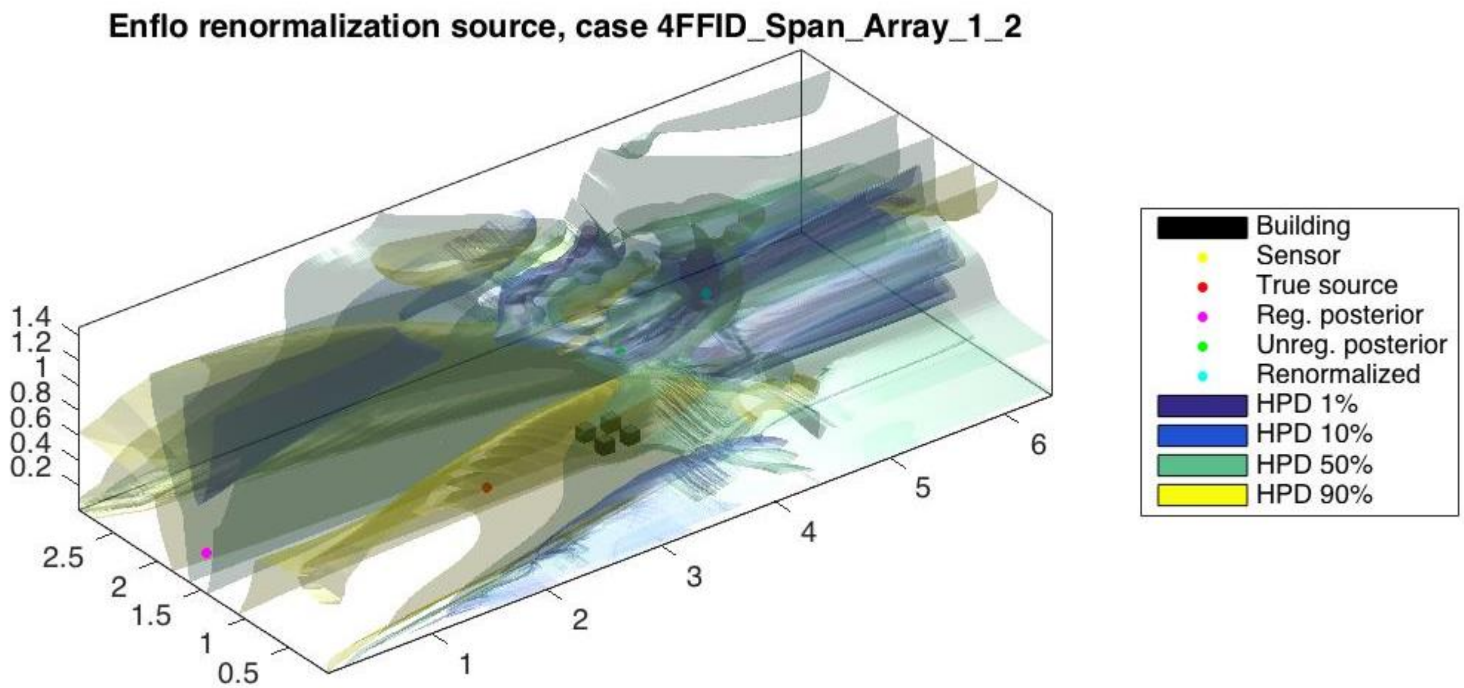
Highest posterior density (HPD) domains



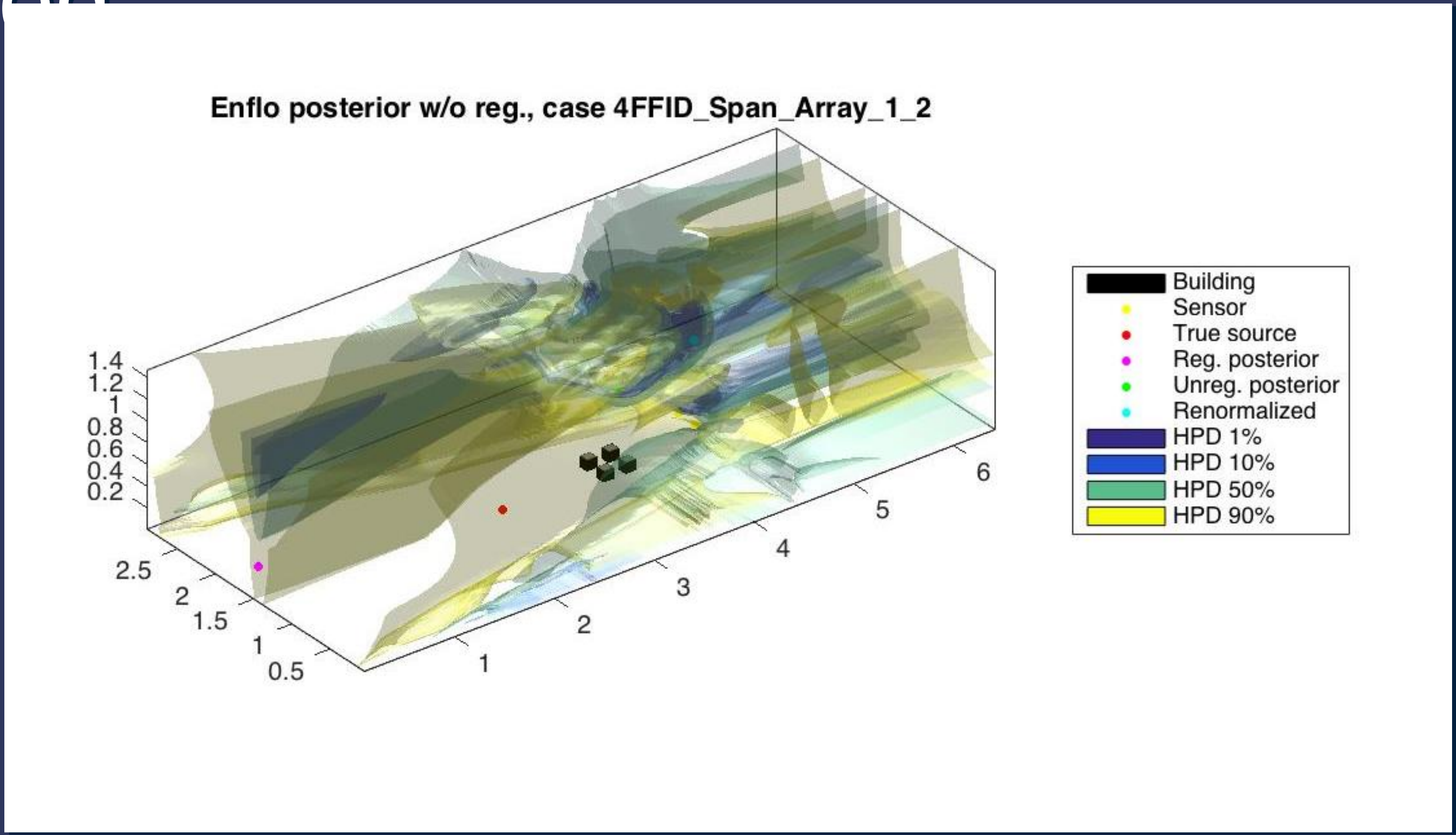
Isosurfaces, 3D posterior w/ reg.



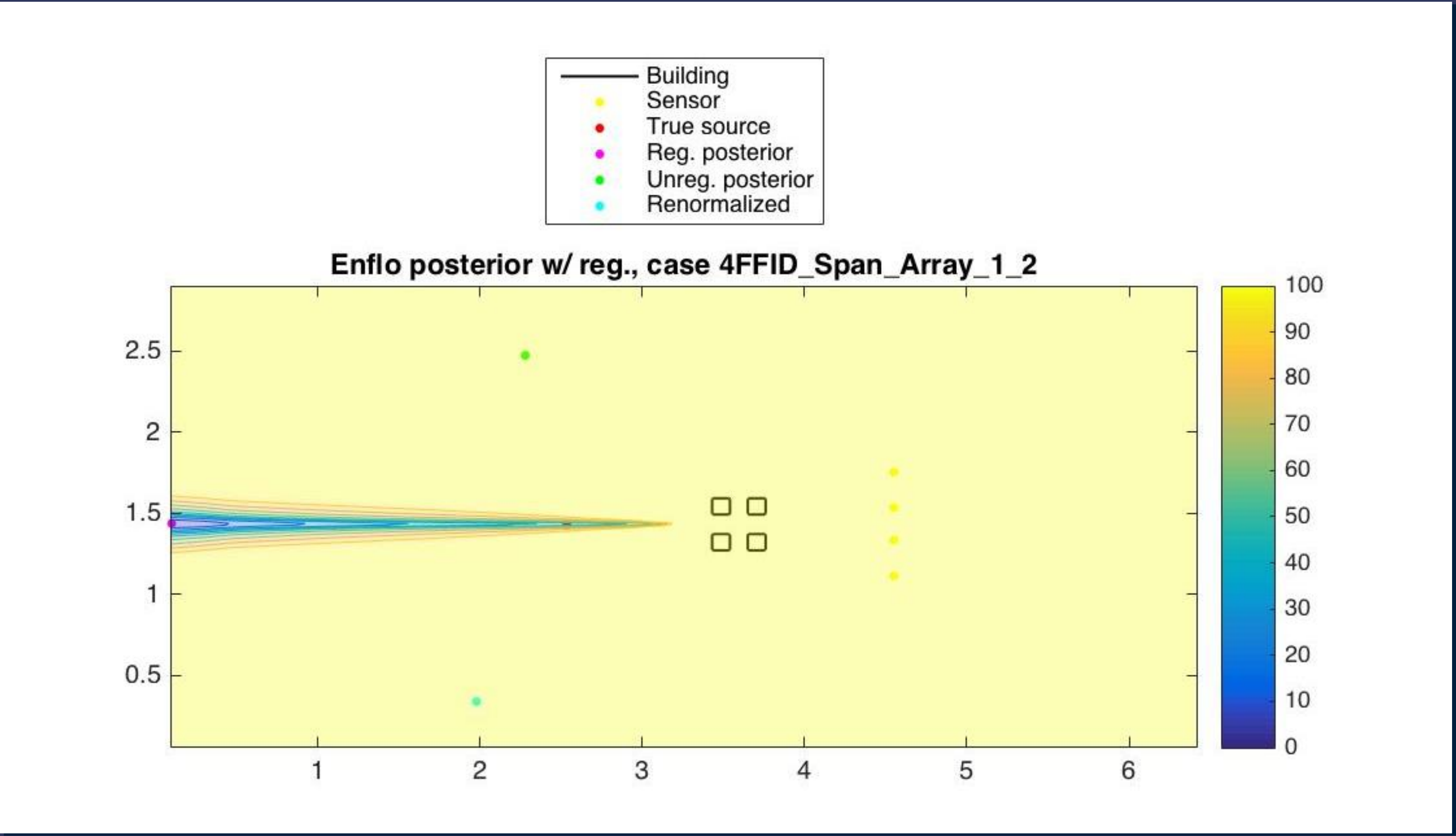
Isosurfaces, 3D renorm. source



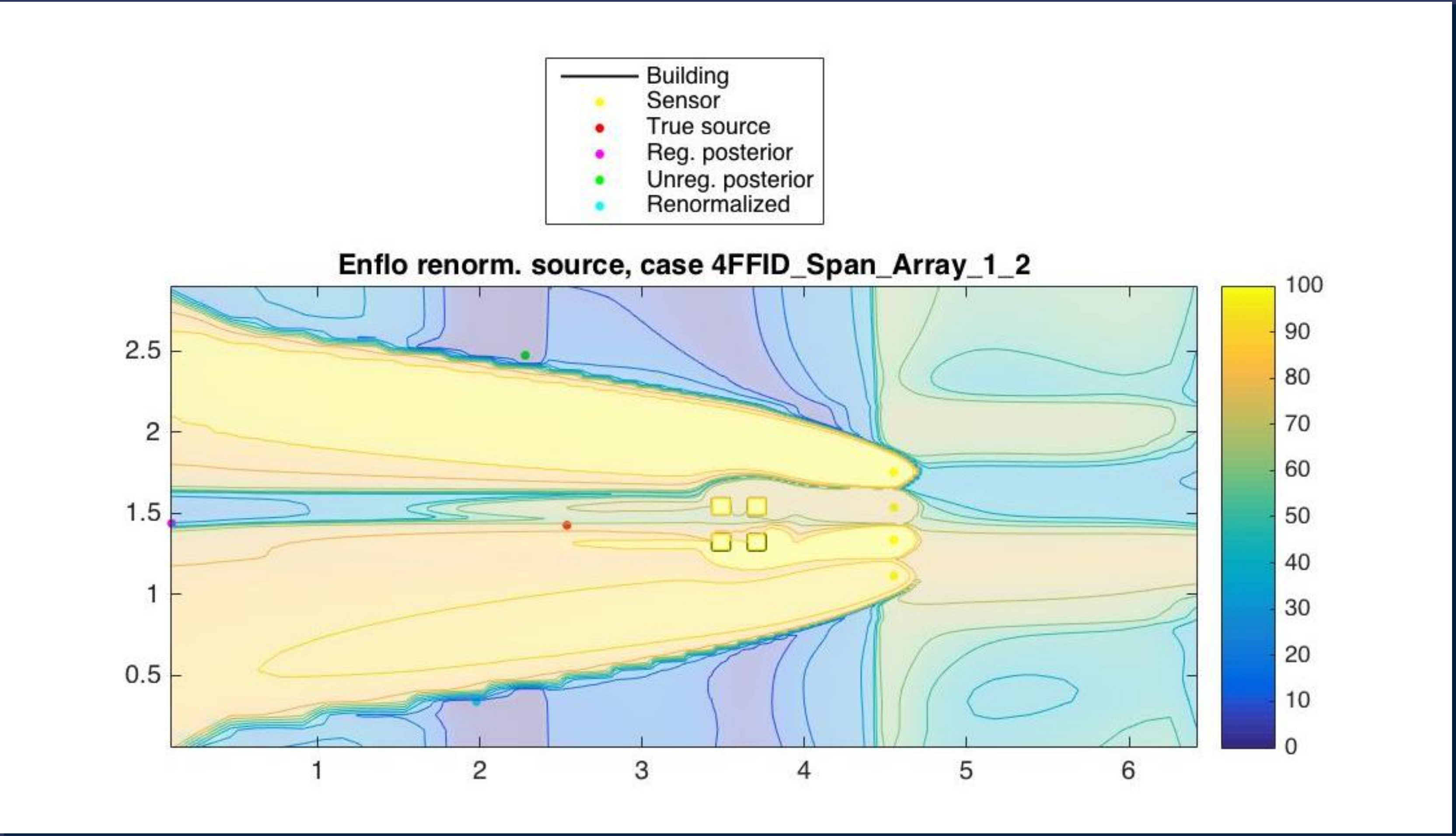
Isosurfaces, 3D posterior w/o reg



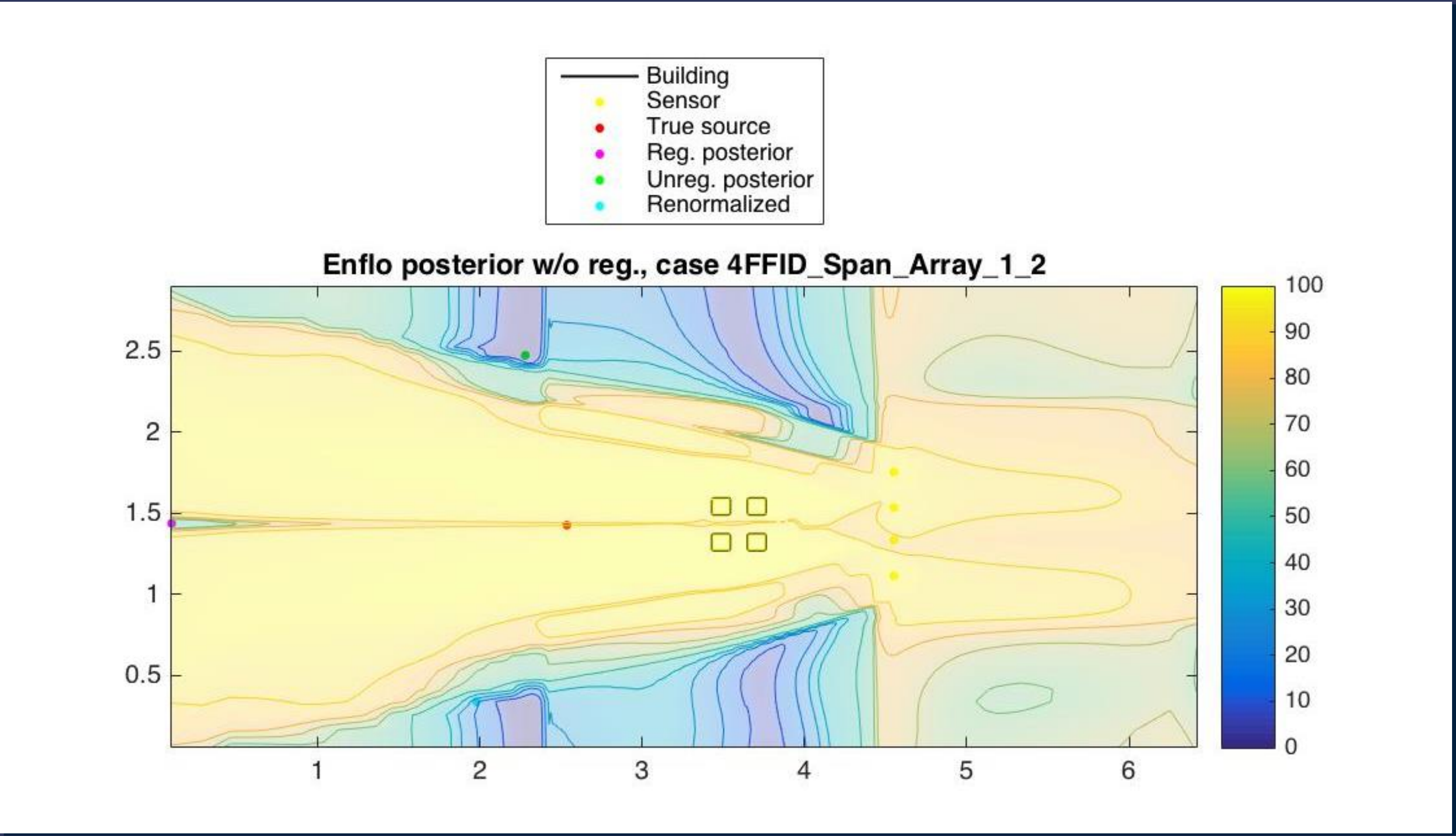
Isocurves, 2D posterior w/ reg.



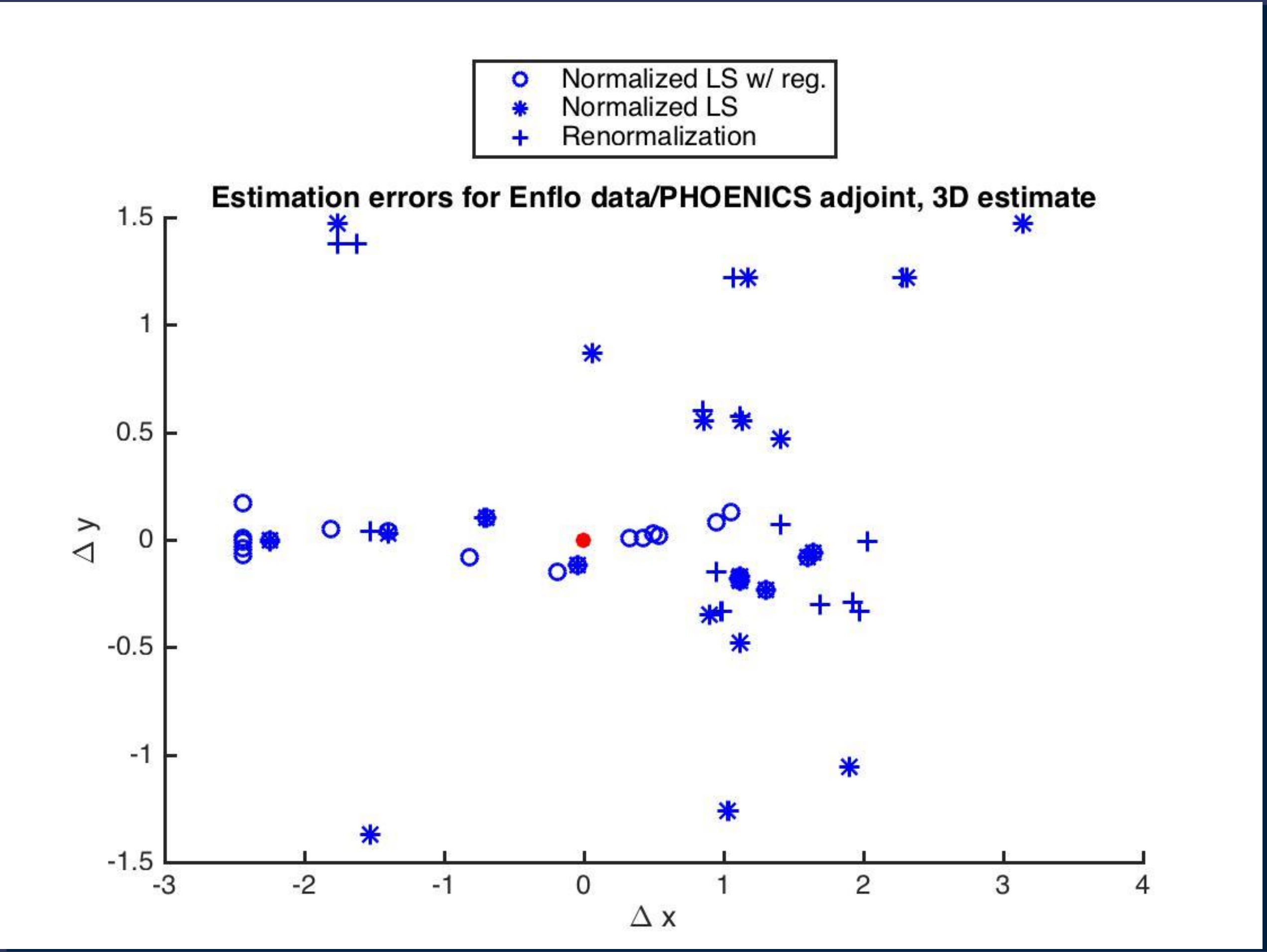
Isocurves, 2D renorm. source



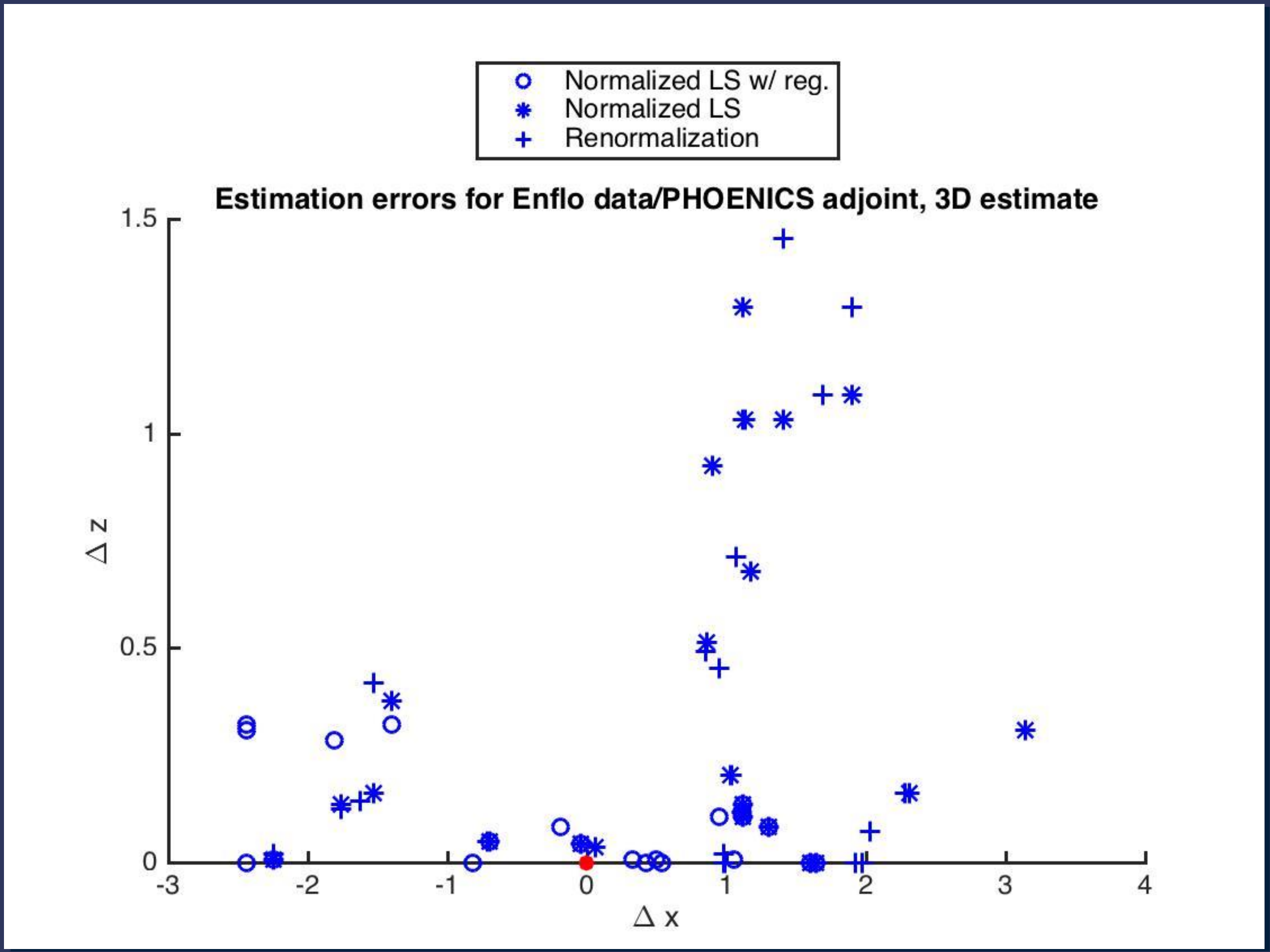
Isocurves, 2D posterior w/o reg.



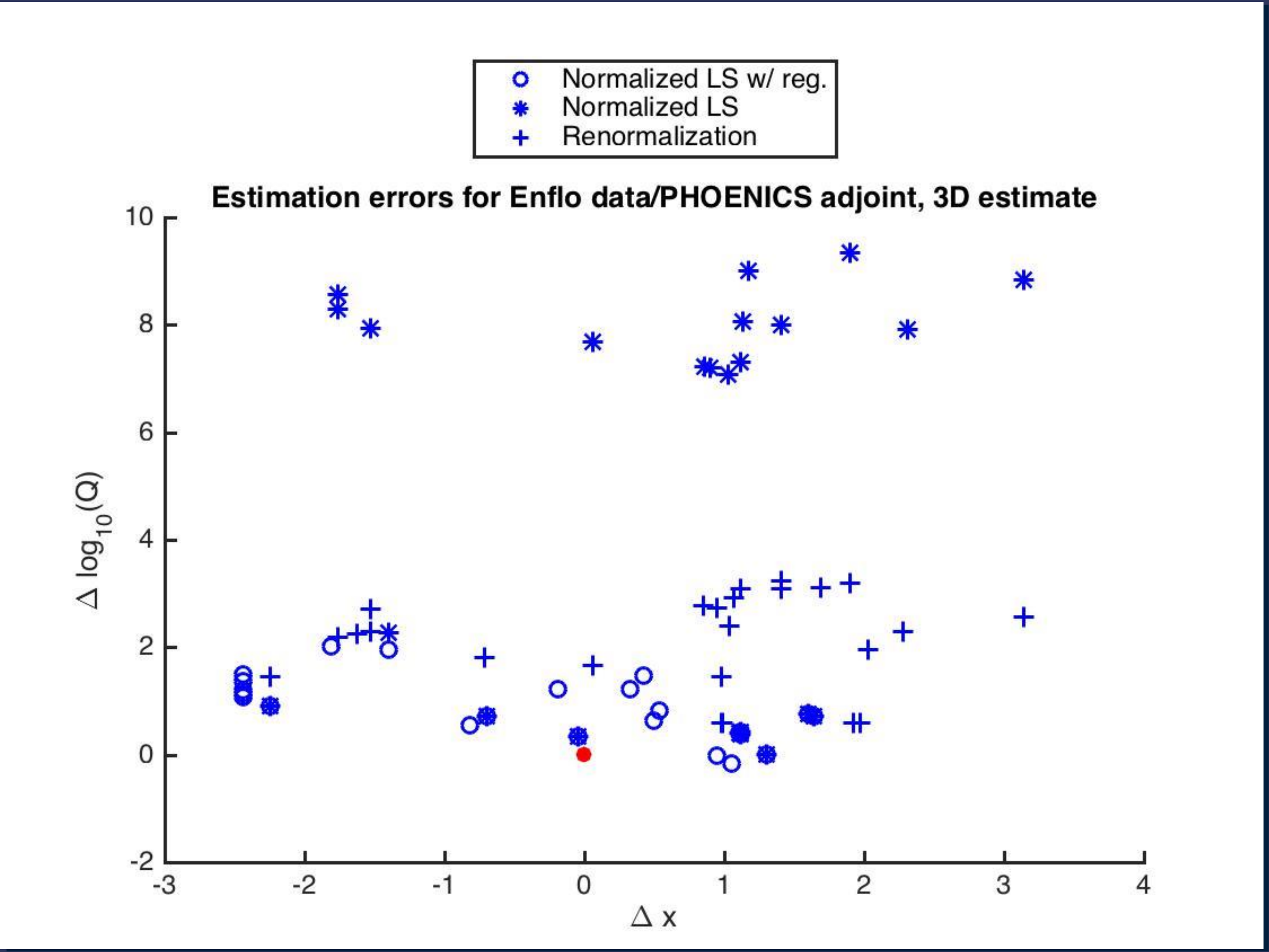
3D comparision, (x,y) errors



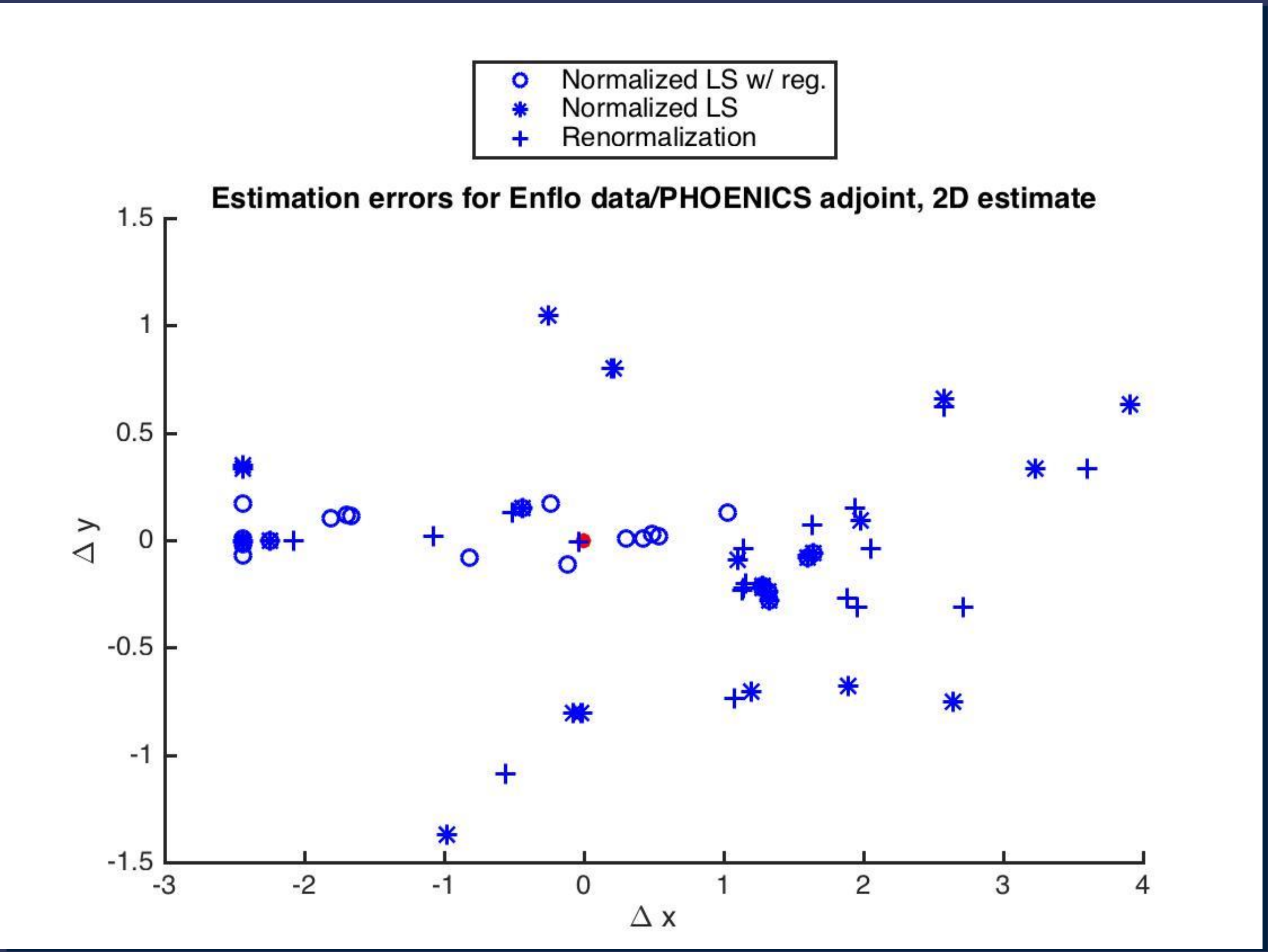
3D comparison, (x,z) errors



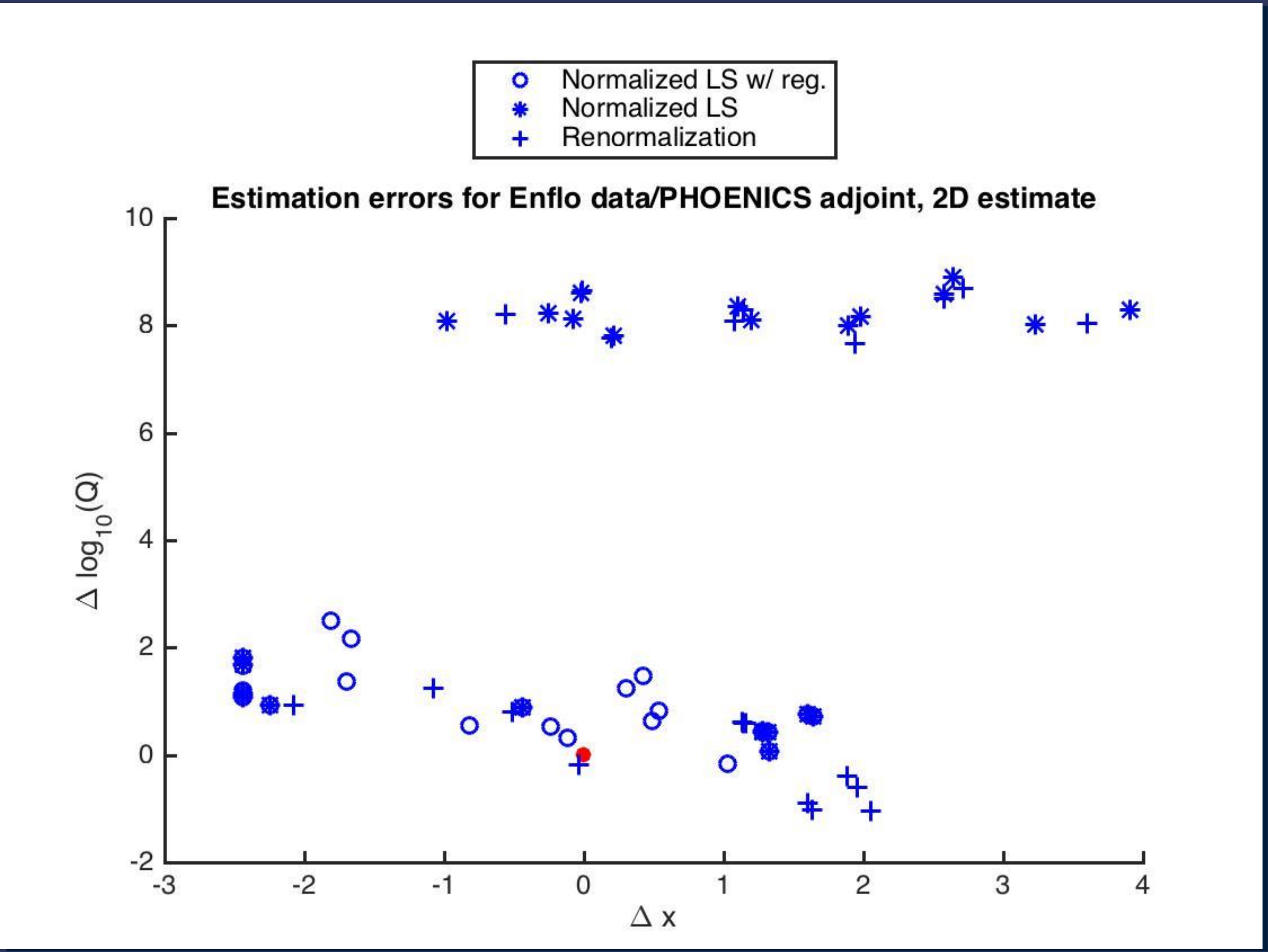
3D comparison, (x,q) errors



2D comparison, (x,y) errors



2D comparison, (*x*,*q*) errors



Mean absolute errors, 24 data sets

3D	LS with reg.	Renormalization	LS
x	1.29	1.47	1.35
y	0.085	0.63	0.62
z	0.083	0.39	0.62
$\log_{10}(q)$	0.87	2.15	4.97

2D	LS with reg.	Renormalization	LS
x	1.35	1.60	1.61
y	0.10	0.32	0.45
$\log_{10}(q)$	0.97	4.17	4.86

FOI

Some conclusions and remarks

- All methods suffers from outliers
- Regularization suppresses outliers in y,z,q but not so much in x
- Lack of regularization gives complicated objective functions and hard optimization problems, reflecting the ill-posedness of the inverse problem
- Implementation has been verified on synthetic data (not presented here)

FOI

Thank you!