particle flow for nonlinear filters

Fred Daum & Jim Huang

26 May 2011

nonlinear filtering problem

$$dx = F(x, t)dt + G(x, t)\,dw$$

continuous time dynamics

x = d-dimensional state vector
t = time
w(t) = process noise vector

discrete time measurements

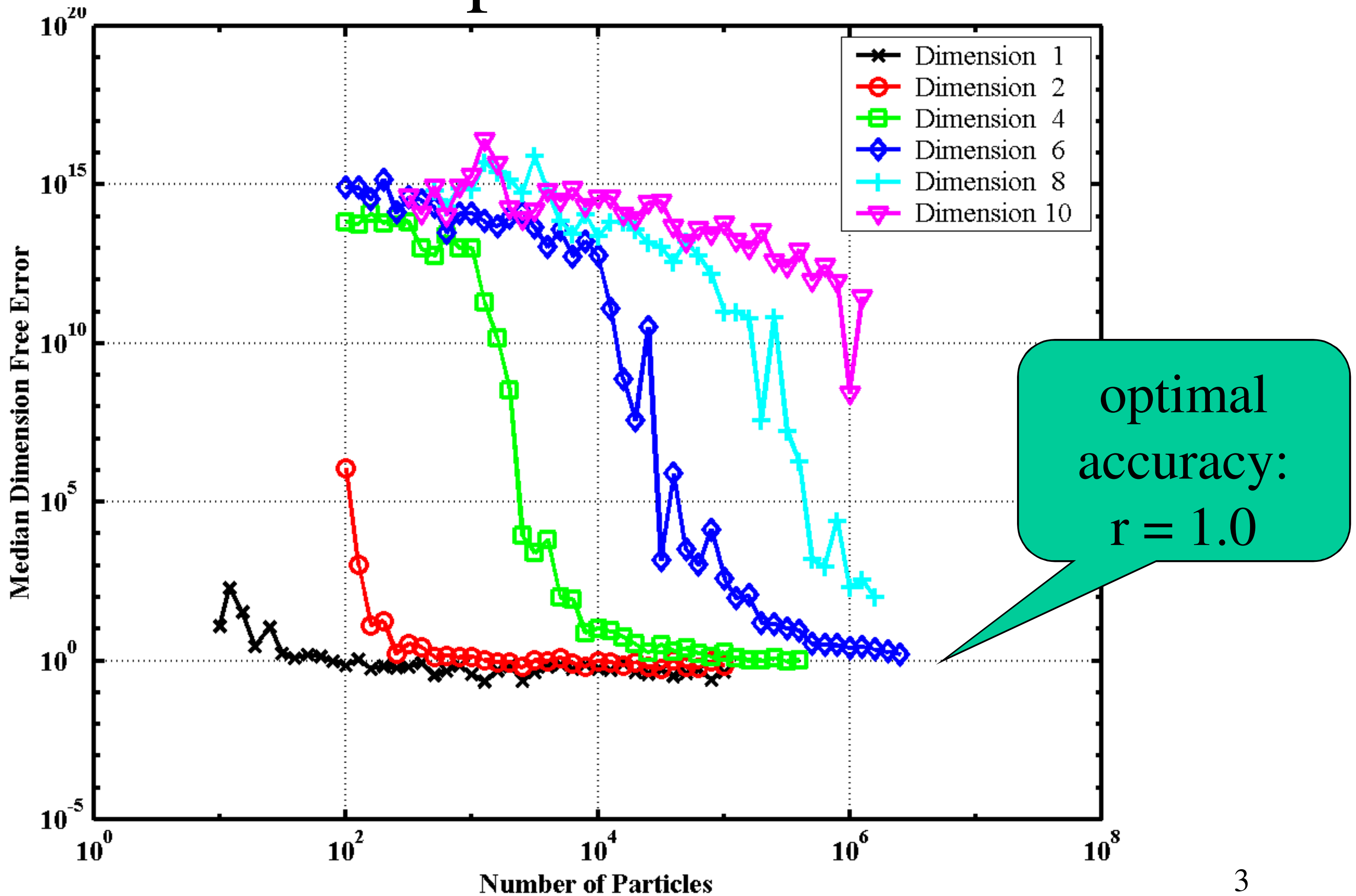
$$z(t_k) = H(x(t_k), t_k, v_k)$$

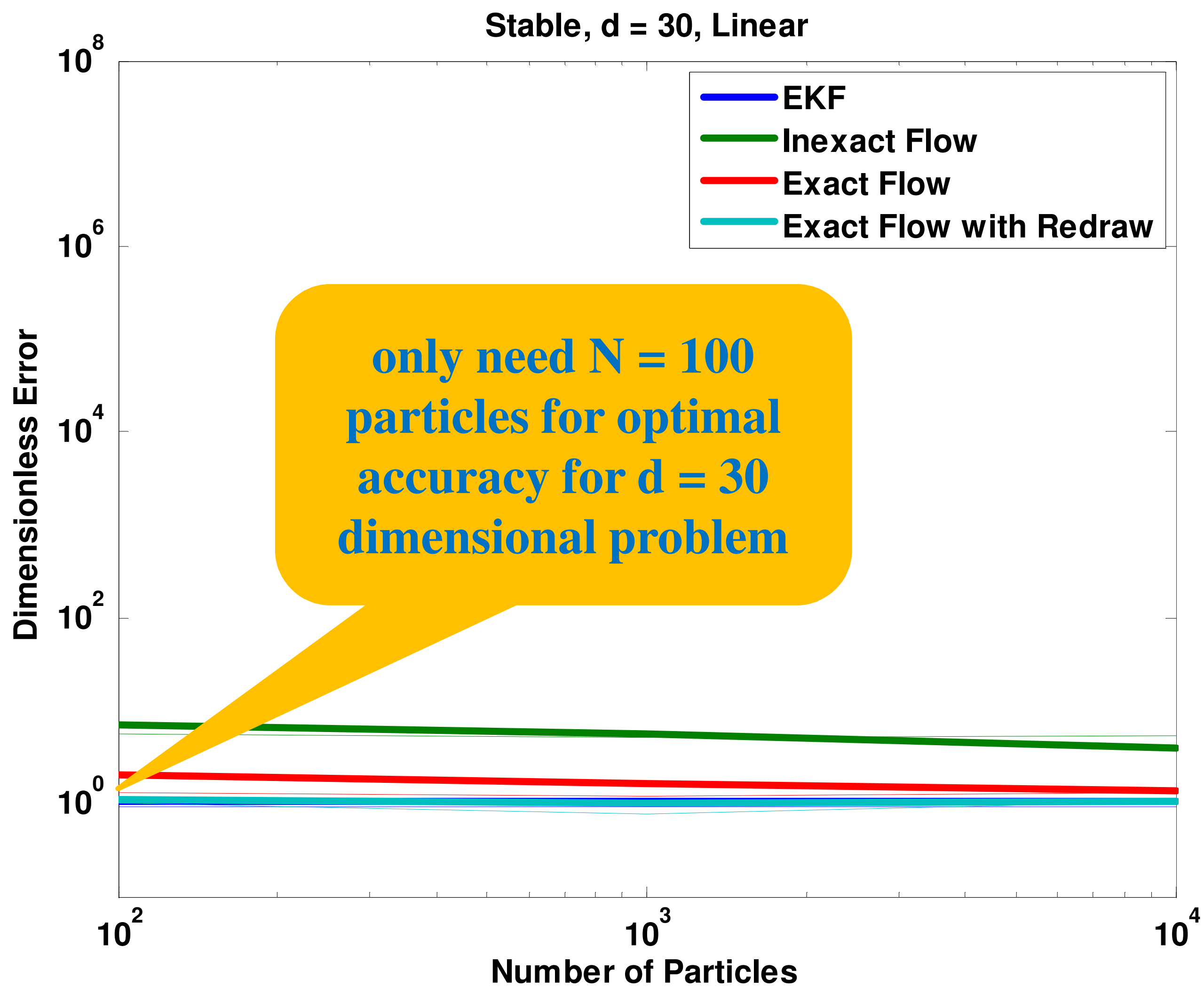
$z(t_k)$ = m-dimensional measurement vector
t_k = time of k^{th} measurement
v_k = measurement noise vector

$p(x, t \mid Z_k)$ = probability density of x at time t given Z_k

Z_k = set of all measurements up to & including time t_k

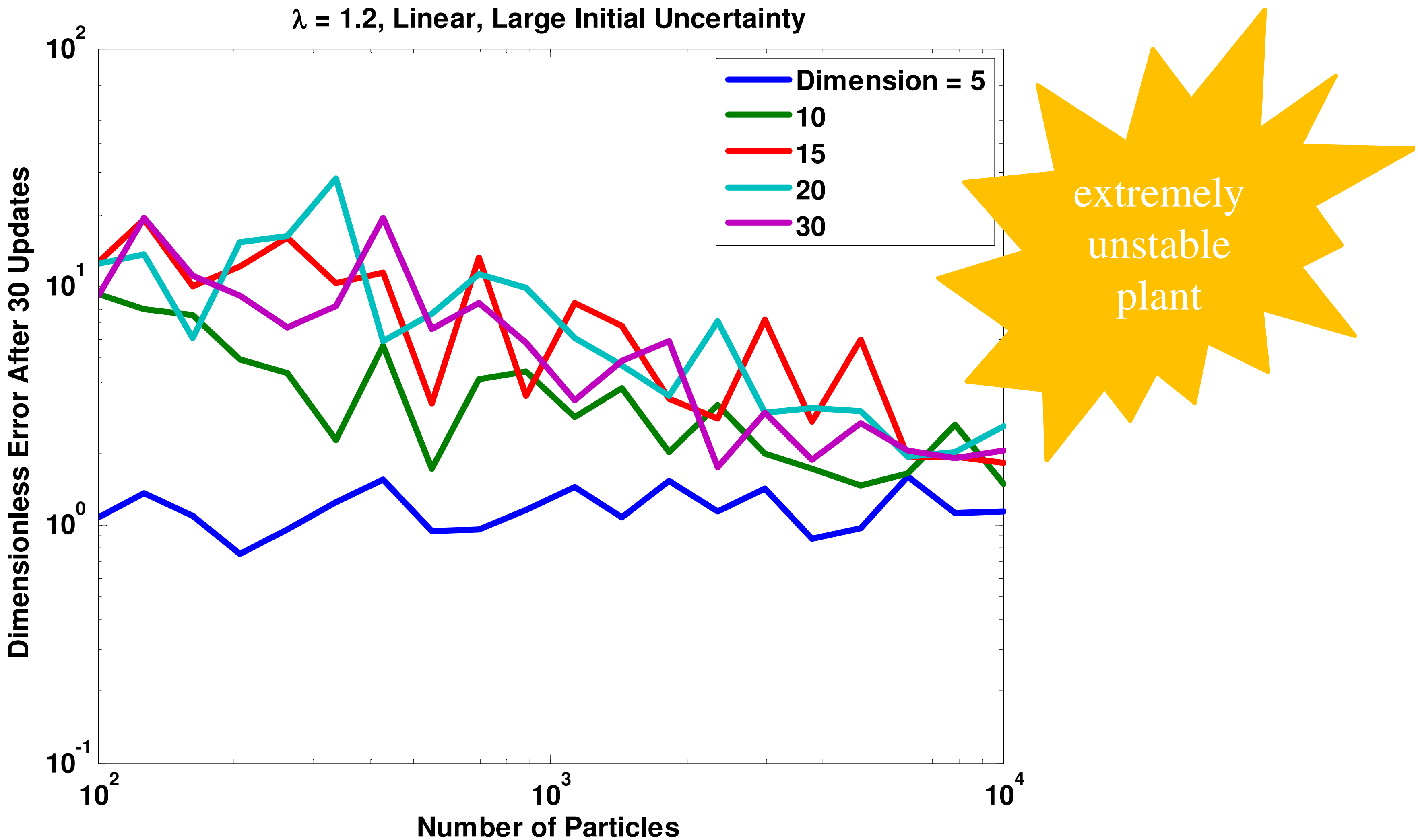
curse of dimensionality for classic particle filter





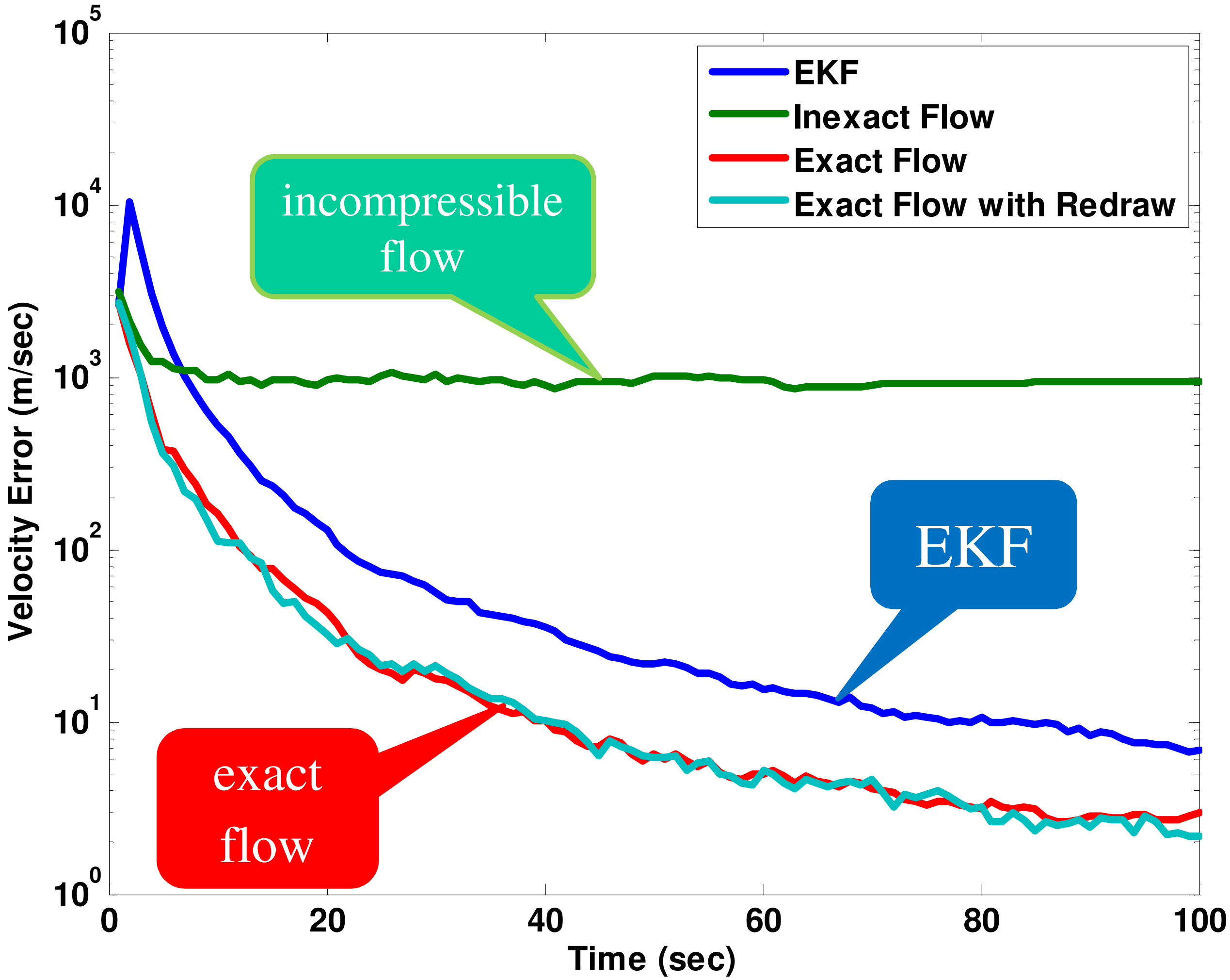
Stable, d = 30, Linear
Dimensionless Error
Number of Particles
EKF
Inexact Flow
Exact Flow
Exact Flow with Redraw
only need N = 100 particles for optimal accuracy for d = 30 dimensional problem

exact flow: performance vs. number of particles

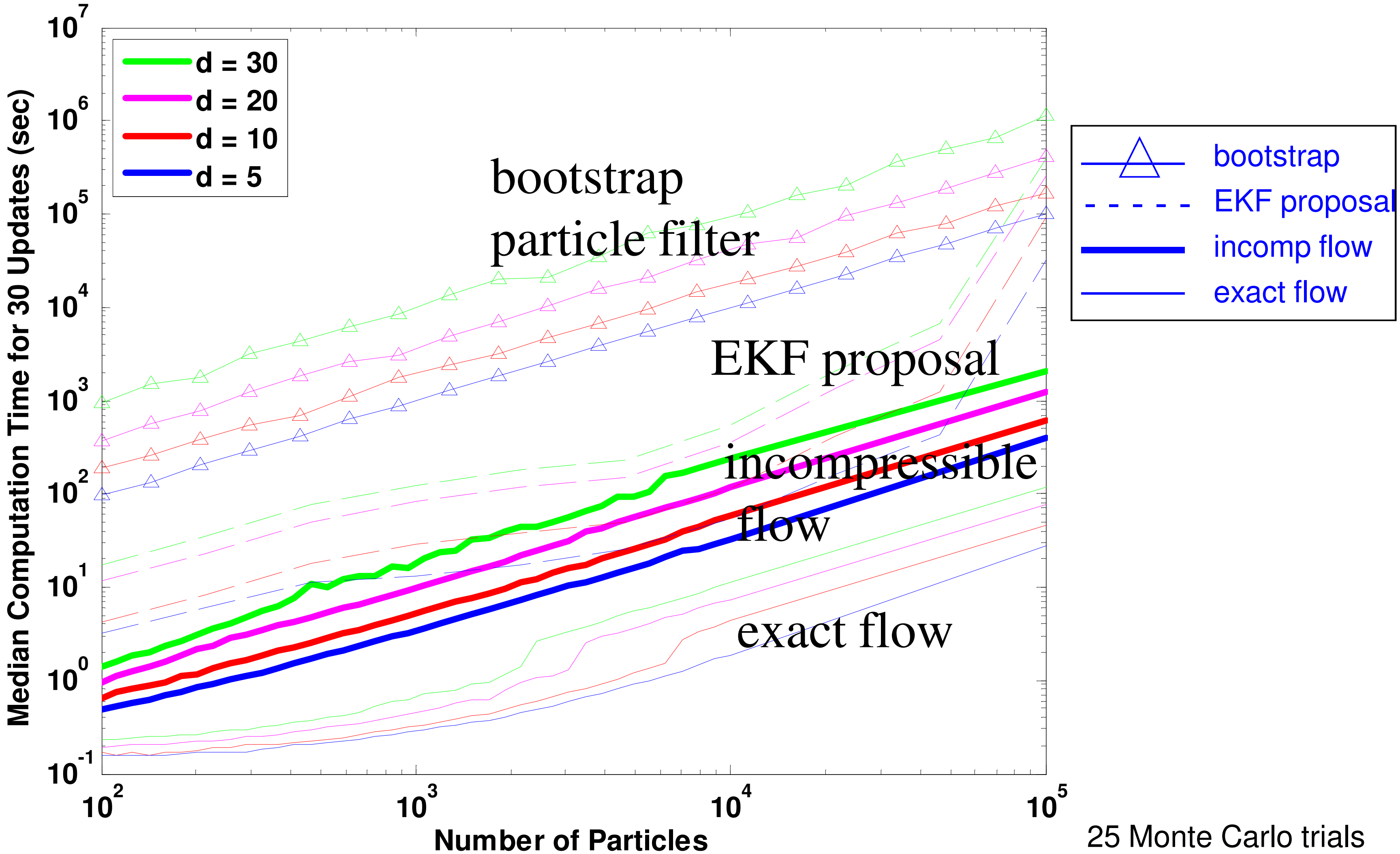


25 Monte Carlo Trials

radar tracking ballistic missile (d =6 & N = 100 particles)



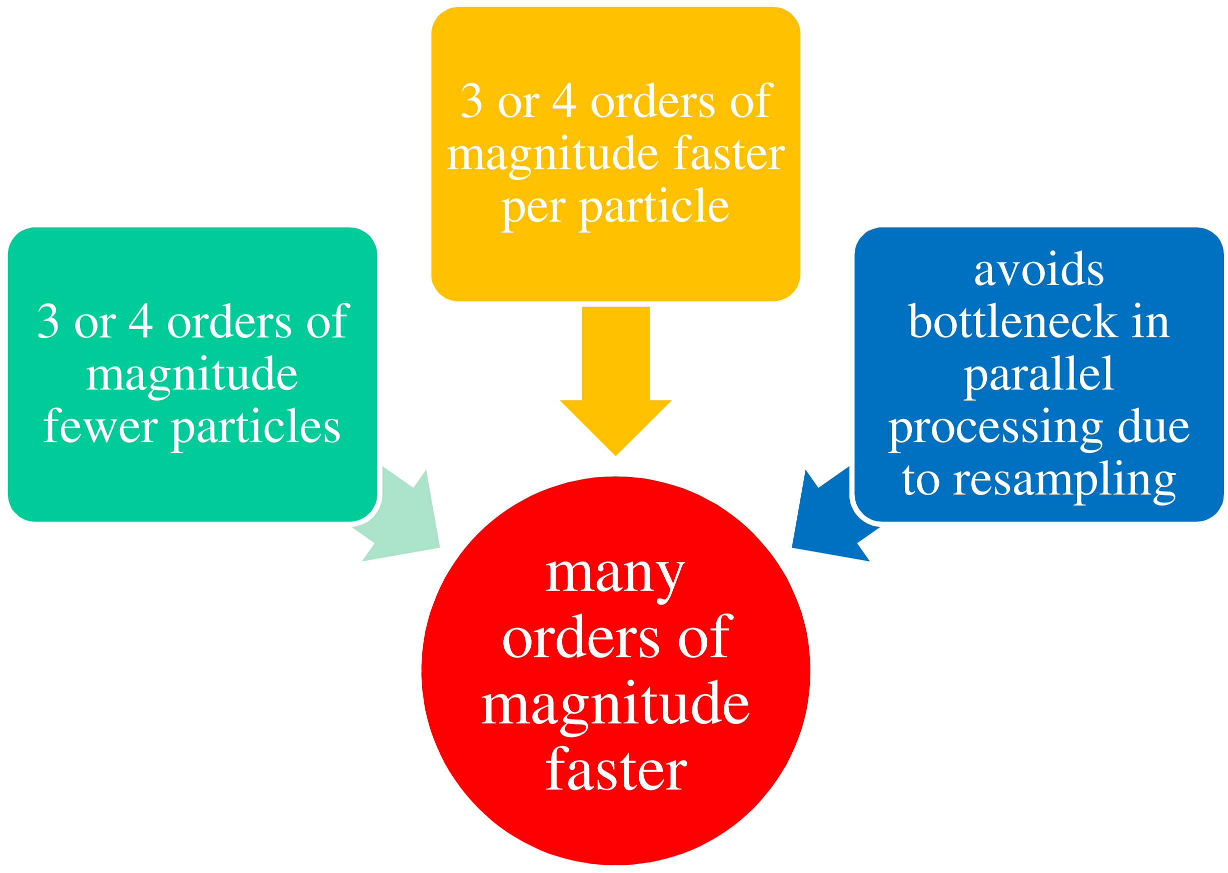
exact flow filter is many orders of magnitude faster per particle than standard particle filters



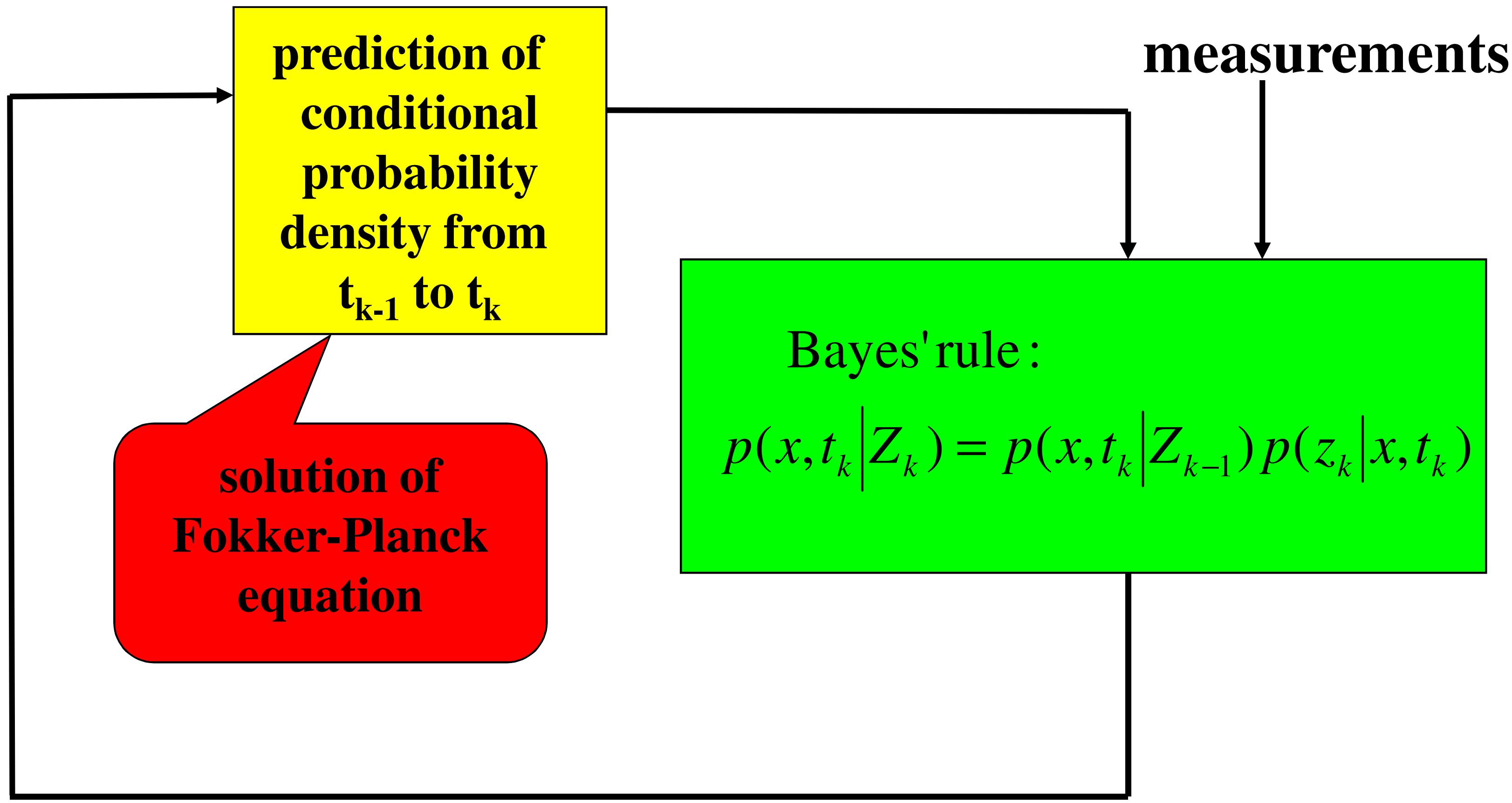
25 Monte Carlo trials

* Intel Corel 2 CPU, 1.86GHz, 0.98GB of RAM, PC-MATLAB version 7.7

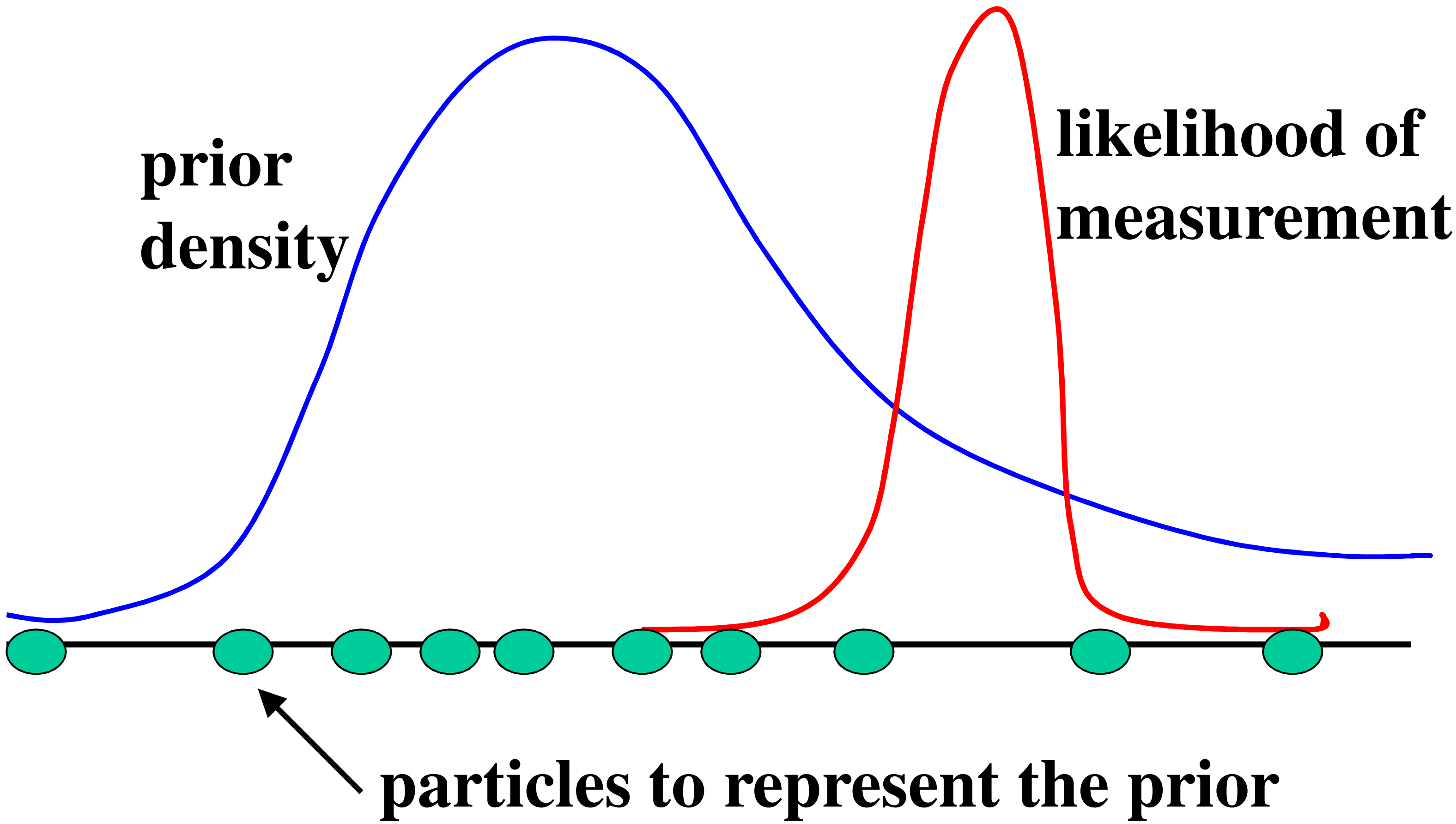
particle flow filter is many orders of magnitude faster real time computation (for the same or better estimation accuracy)



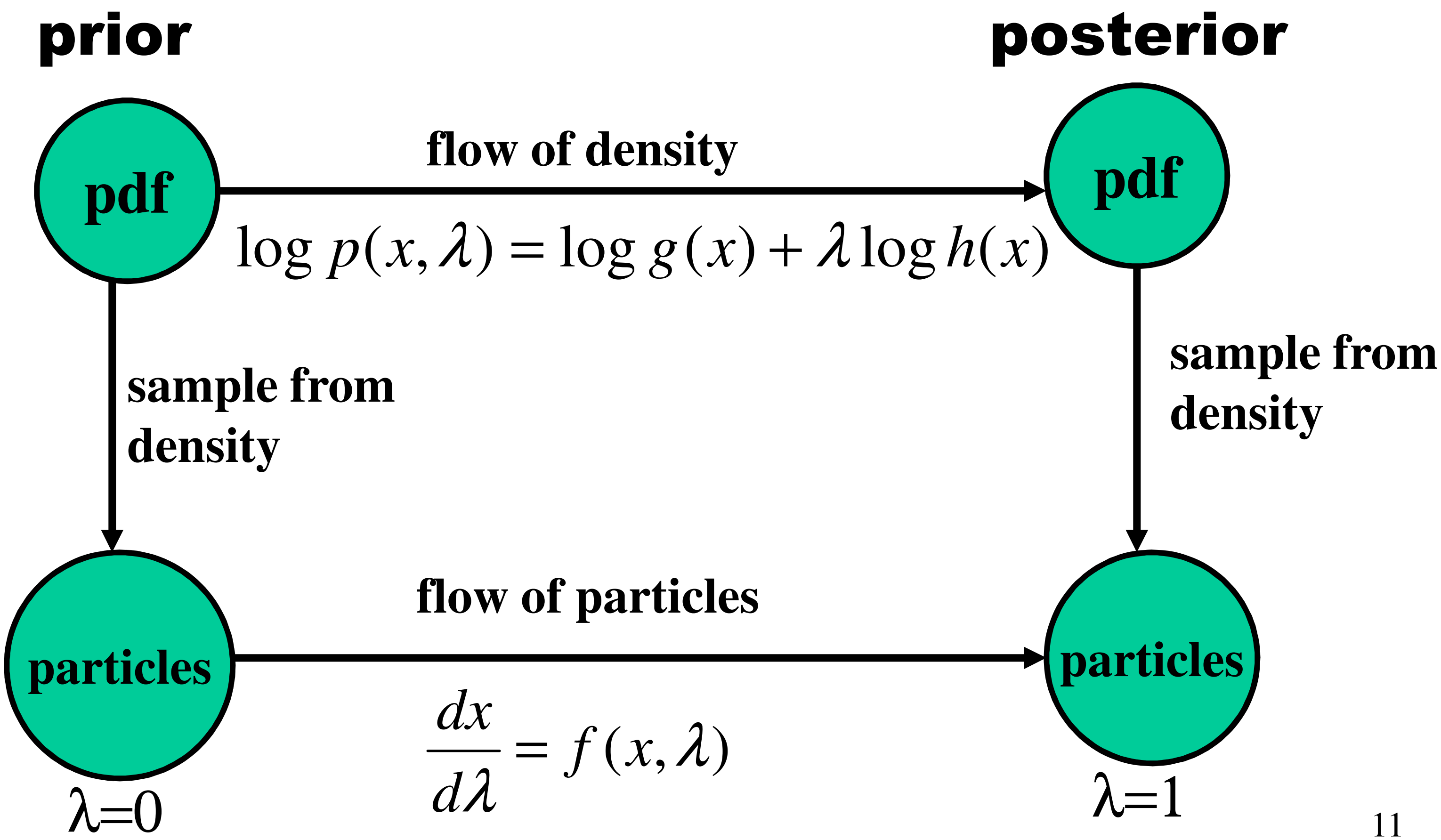
nonlinear filter



particle degeneracy



induced flow of particles for Bayes' rule



linear first order highly underdetermined PDE:

$$div(q(x,\lambda)) = Tr\left[\frac{\partial q(x,\lambda)}{\partial x}\right] = \eta(x,\lambda)$$

$$\eta(x,\lambda) = -p(x,\lambda)\log h(x)$$

$$\eta = \frac{\partial q_1}{\partial x_1} + \frac{\partial q_2}{\partial x_2} + \ldots + \frac{\partial q_d}{\partial x_d}$$

like Gauss' divergence law in electromagnetics

function values are only known at random points in d-dimensional space

q = pf
f = unknown function
p & η = known at random points

We want dx/dλ = f(x,λ) to be a stable dynamical system.

method to solve PDE	how to pick unique solution	computation
1. generalized inverse of linear differential operator	**minimum norm***	**fast Poisson solver in d-dimensions or Coulomb's law or other**
2. Poisson's equation	**gradient of potential* (assume irrotational flow)**	**fast Poisson solver in d-dimensions or Coulomb's law or other**
3. generalized inverse of gradient of log-homotopy	**assume incompressible flow (i.e., divergence free flow)**	**fast (but need to compute the gradient from random points)**
4. most general solution	**most robustly stable filter or random pick, etc.**	**fast (but need to compute the gradient from random points)**
5. separation of variables (Gaussian)	**pick solution of specific form (polynomial)**	**extremely fast (formula)**
6. separation of variables (exponential family)	**pick solution of specific form (finite basis functions)**	**very fast (formula)**
7. variational formulation (Gauss & Hertz)	**convex function minimization**	**ODEs**
8. optimal control formulation	**convex functional minimization**	**Euler-Lagrange PDEs (or maybe ODES for nice problem)**
9. direct integration (of first order linear PDE in divergence form)	**choice of d-1 arbitrary functions**	**one-dimensional integral**
10. generalized method of characteristics	**more conditions (e.g., curl = given & chain rule)**	**ODEs from chain rule**
11. another homotopy (inspired by Gromov's h-principle)	**initial condition of ODE & uniqueness of sol. to ODE**	**ODEs from homotopy**
12. finite dimensional parametric flow (e.g., f = Ax+b)	**non-singular matrix to invert**	**d^3 or d^6 (least squares for d or d^2 parameters, i.e., A & b)**
13. Fourier transform of PDE (divergence form of PDE has constant coefficients)	**minimum norm* or most stable flow**	**Gaussian sum makes inverse very fast (by inspection)**

exact particle flow for Gaussian densities:

$$\frac{dx}{d\lambda} = f(x, \lambda)$$

$$\log(h) = -div(f) - \frac{\partial \log p}{\partial x} f$$

for g & h Gaussian, we can solve for f exactly:

$$f = Ax + b$$

$$A = -\frac{1}{2} PH^T \left[\lambda HPH^T + R\right]^{-1} H$$

$$b = (I + 2\lambda A)\left[(I + \lambda A)PH^T R^{-1} z + A\bar{x}\right]$$

automatically stable under very mild conditions & extremely fast

effect of divergence of f:

$$\frac{dx}{d\lambda} = f(x,\lambda)$$

$$\log(h) = -div(f) - \frac{\partial \log p}{\partial x} f$$

suppose that div(f) is given

$$f = -\left[\frac{\partial \log p}{\partial x}\right]^{\#} \left[\log h + div(f)\right]$$

if div(f) = 0 we get incompressible flow

effect of div(f) is to change the speed of particle flow

incompressible particle flow

$$\frac{\mathrm{dx}}{\mathrm{d}\lambda} = {-\log(h)\left(\frac{\partial \log p}{\partial x}\right)^T} \Big/ {\left\|\frac{\partial \log p}{\partial x}\right\|^2}$$

$$\frac{dx}{d\lambda} = 0 \qquad \text{for zero gradient}$$

solving Poisson's equation:

$$\frac{dx}{d\lambda} = f(x,\lambda) = \left[\frac{\partial V(x,\lambda)}{\partial x}\right]^T / \, p(x,\lambda)$$

$$Tr\left[\frac{\partial^2 V(x,\lambda)}{\partial x^2}\right] = -\log h(x)\, p(x,\lambda)$$

Poisson's equation

$$V(x,\lambda) = \int \log h(y)\, p(y,\lambda) \frac{c}{\|x-y\|^{d-2}} dy$$

in which

$$\mathrm{c} = \Gamma(\frac{\mathrm{d}}{2} - 1) / 4\pi^{d/2}$$

integration by parts yields:

$$\frac{\partial \mathrm{V}(\mathrm{x},\lambda)}{\partial \mathrm{x}} = -\int \frac{\partial \log h(y)\, p(y,\lambda)}{\partial y} \frac{c}{\|x-y\|^{d-2}} dy$$

or without integration by parts:

$$\frac{\partial \mathrm{V}(\mathrm{x},\lambda)}{\partial \mathrm{x}} = \int \log h(y)\, p(y,\lambda) \frac{c(2-d)(x-y)^T}{\|x-y\|^{d}} dy$$

d-dimensional Coulomb's law:

for $d \geq 3$

$$\frac{\partial \mathrm{V}(\mathrm{x},\lambda)}{\partial \mathrm{x}} = \int \log h(y)\, p(y,\lambda) \frac{c(2-d)(x-y)^T}{\|x-y\|^d} dy$$

$$\frac{\partial V(x,\lambda)}{\partial x} = E\left[\log h(y) c(2-d)(x-y)^T / \|x-y\|^d\right]$$

$$\frac{\partial V(x_i,\lambda)}{\partial x} \approx \sum_{j \in S_i} \log h(x_j)\, p(x_j,\lambda) \frac{c(2-d)(x_i - x_j)^T}{\|x_i - x_j\|^d}$$

like Coulomb's law

in which $\mathrm{S_i}$ is the set of k - nearest neighbors to the $\mathrm{i^{th}}$ particle

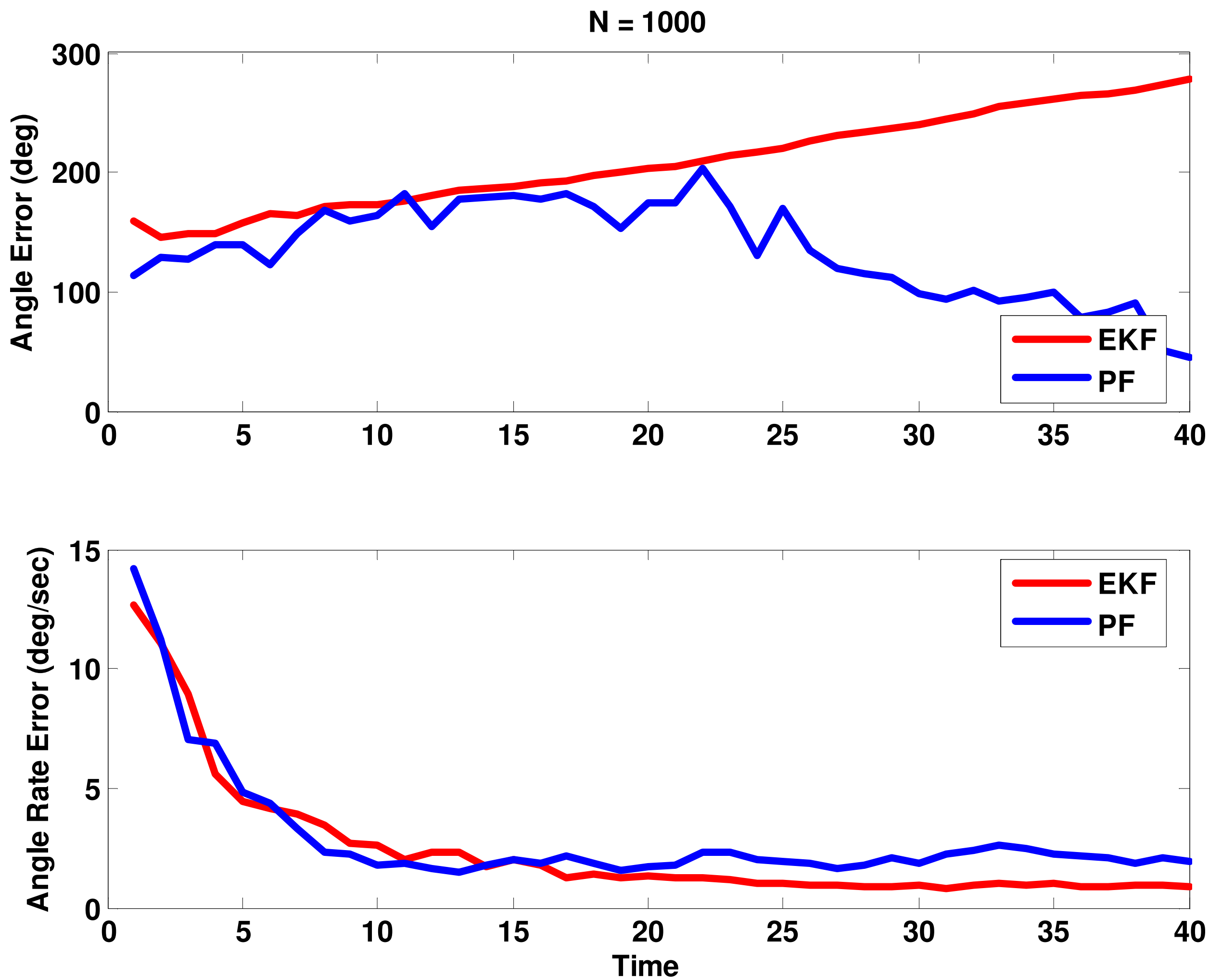
$\mathrm{x_i} = i^{th}$ particle

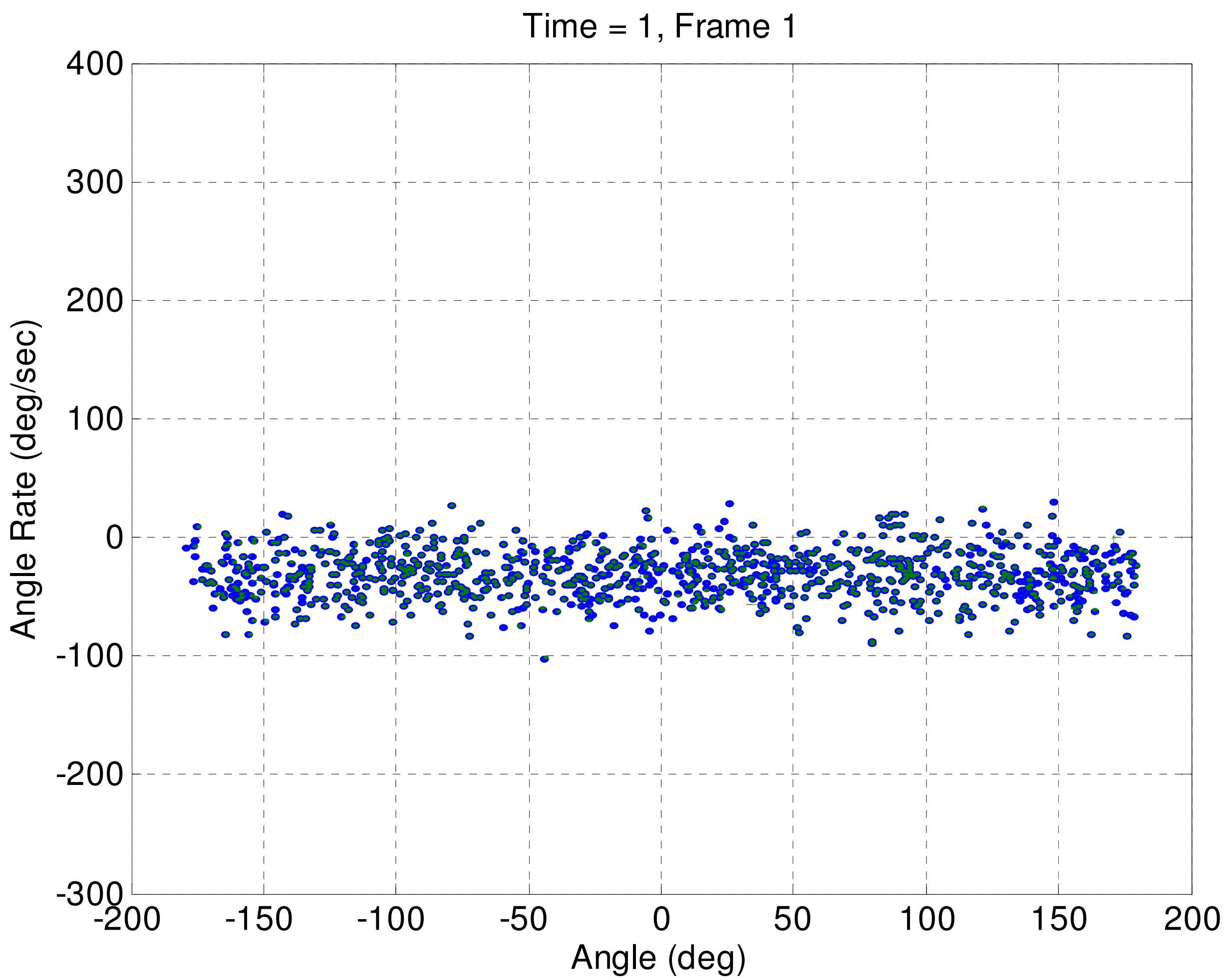
where E(.) denotes the expected value wrt the probability density p(y).

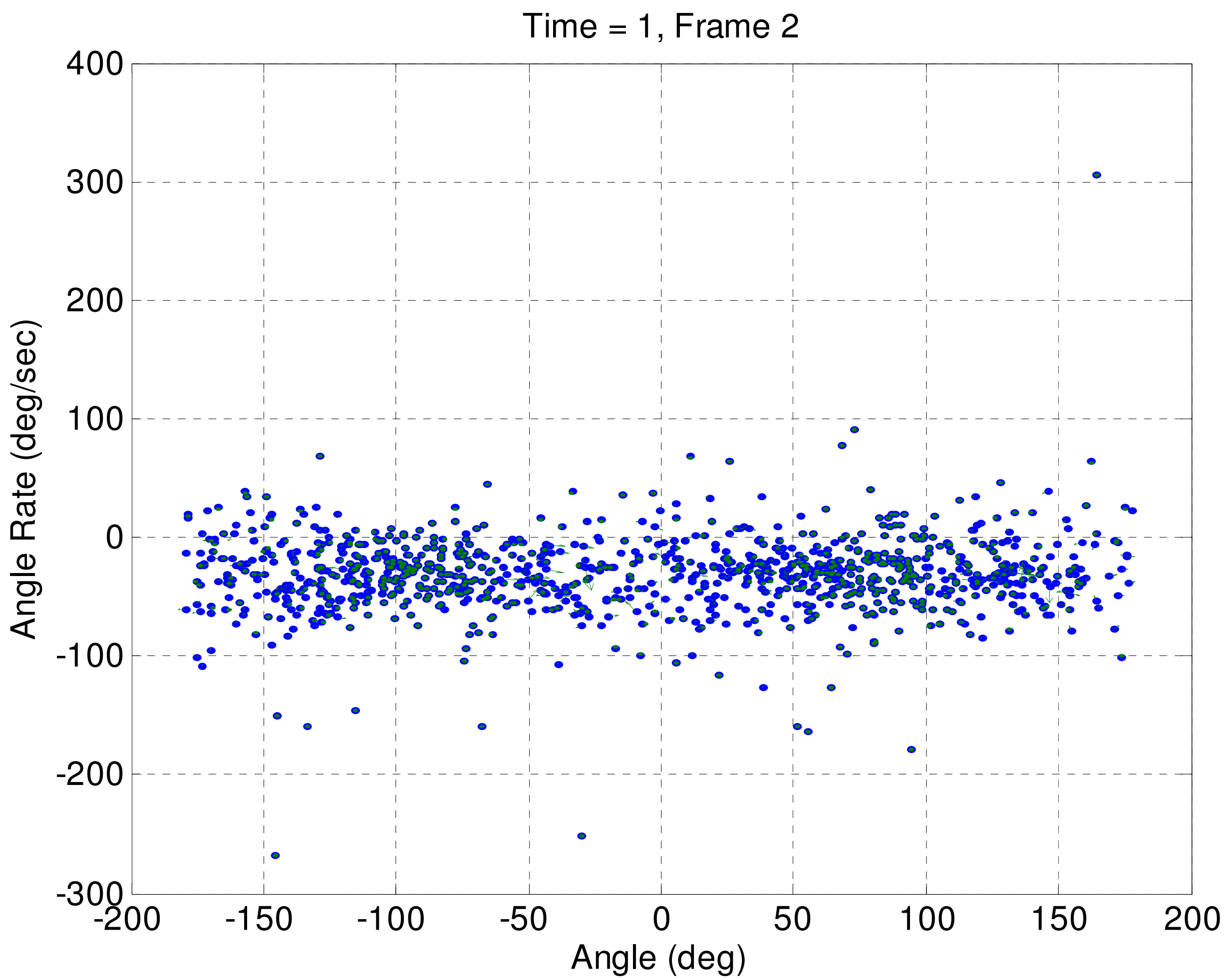
(1) Flavia Lanzara , Vladimir Maz'ya & Gunther Schmidt "On the fast computation of high dimensional volume potentials" arXiv:0911.0443v1 [math.NA] 2 Nov 2009

note: linear computational complexity in d for uniform grid, and it can be extended to scattered data!

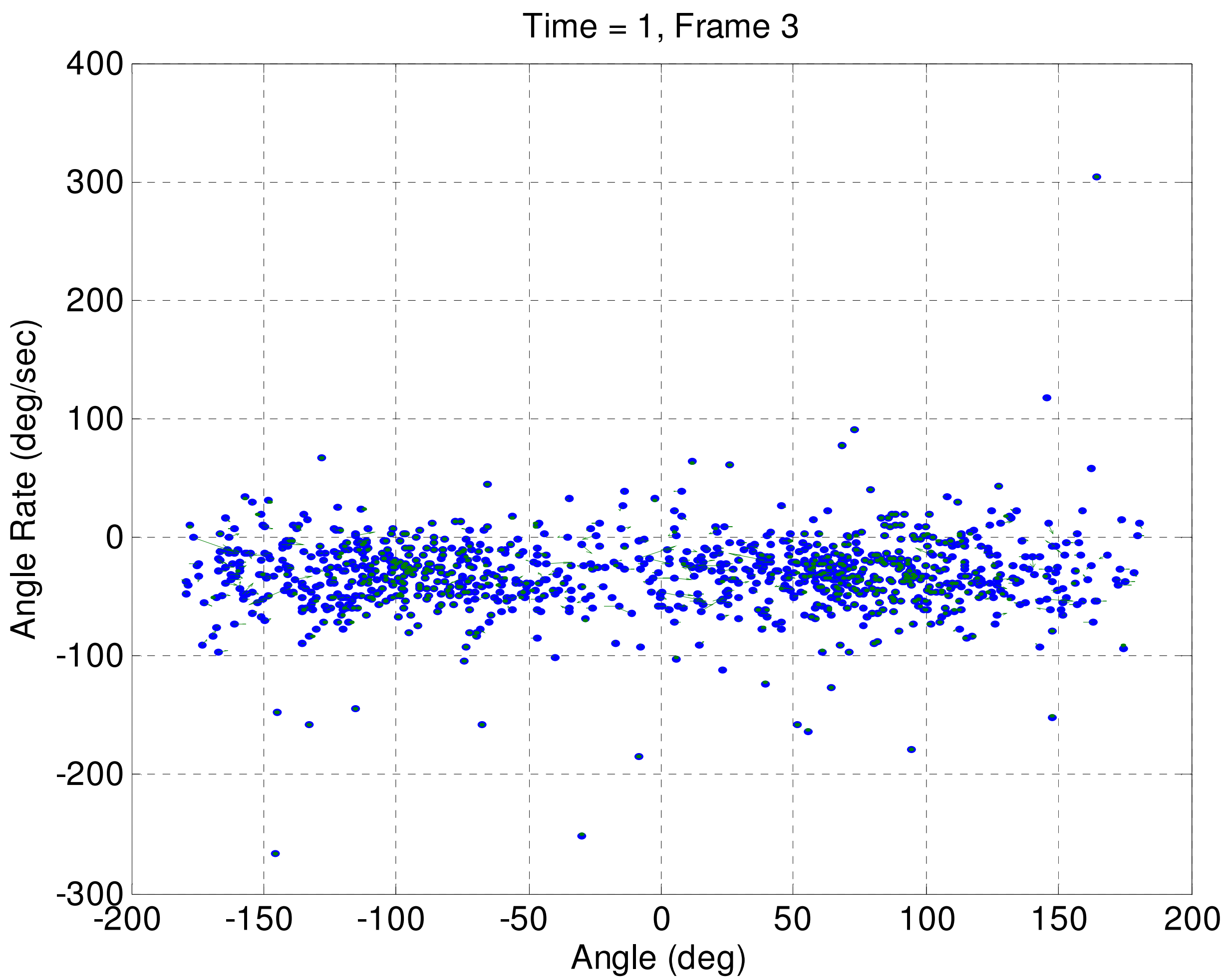
(2) huge literature on fast Poisson solvers (e.g., FMM Rokhlin, Beylkin, Coifman, Hackbush, et al.)


N = 1000
Angle Error (deg)
300
200
100
0
0
5
10
15
20
25
30
35
40
EKF
PF
Angle Rate Error (deg/sec)
15
10
5
0
Time

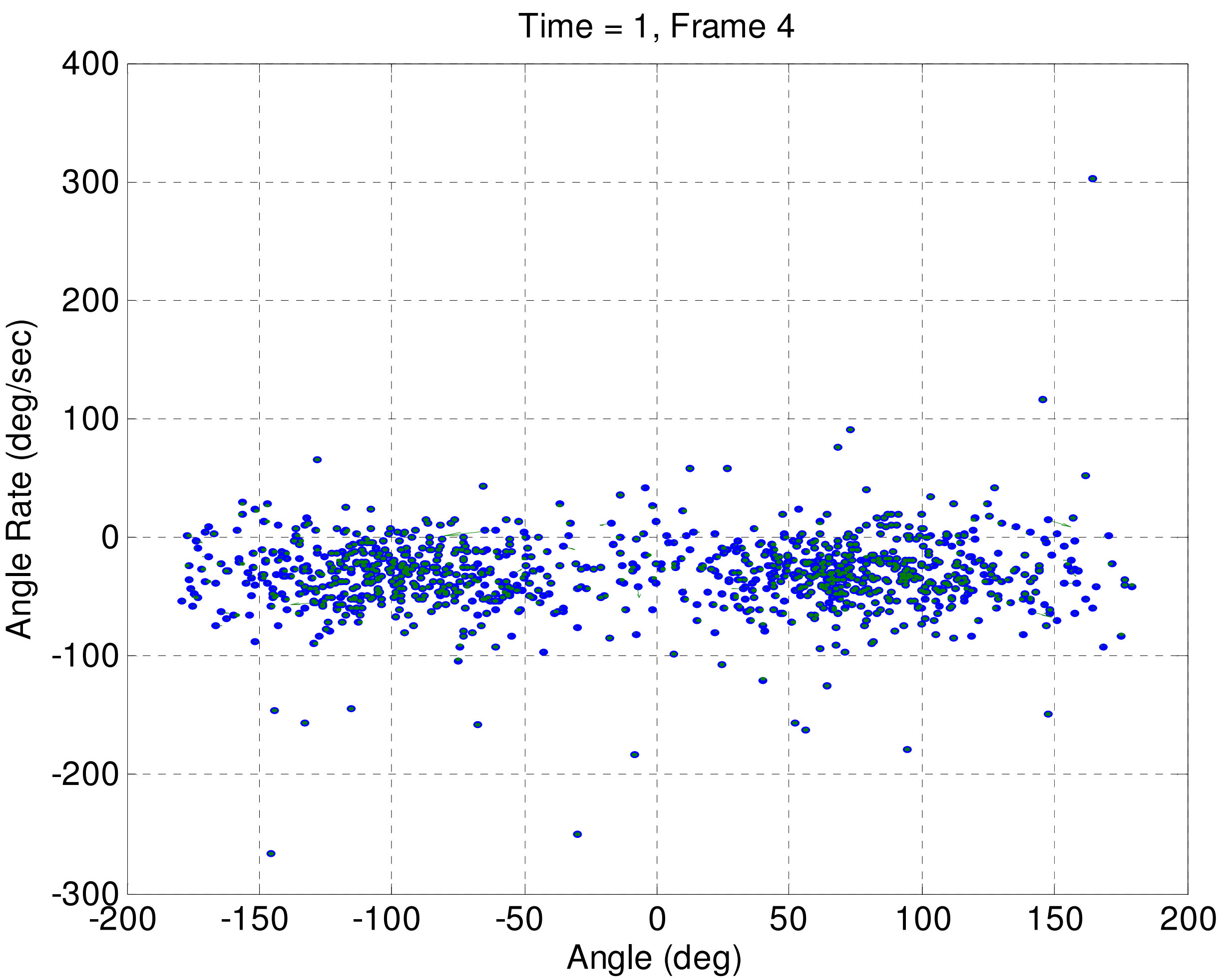

Time = 1, Frame 1
Angle Rate (deg/sec)
Angle (deg)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200

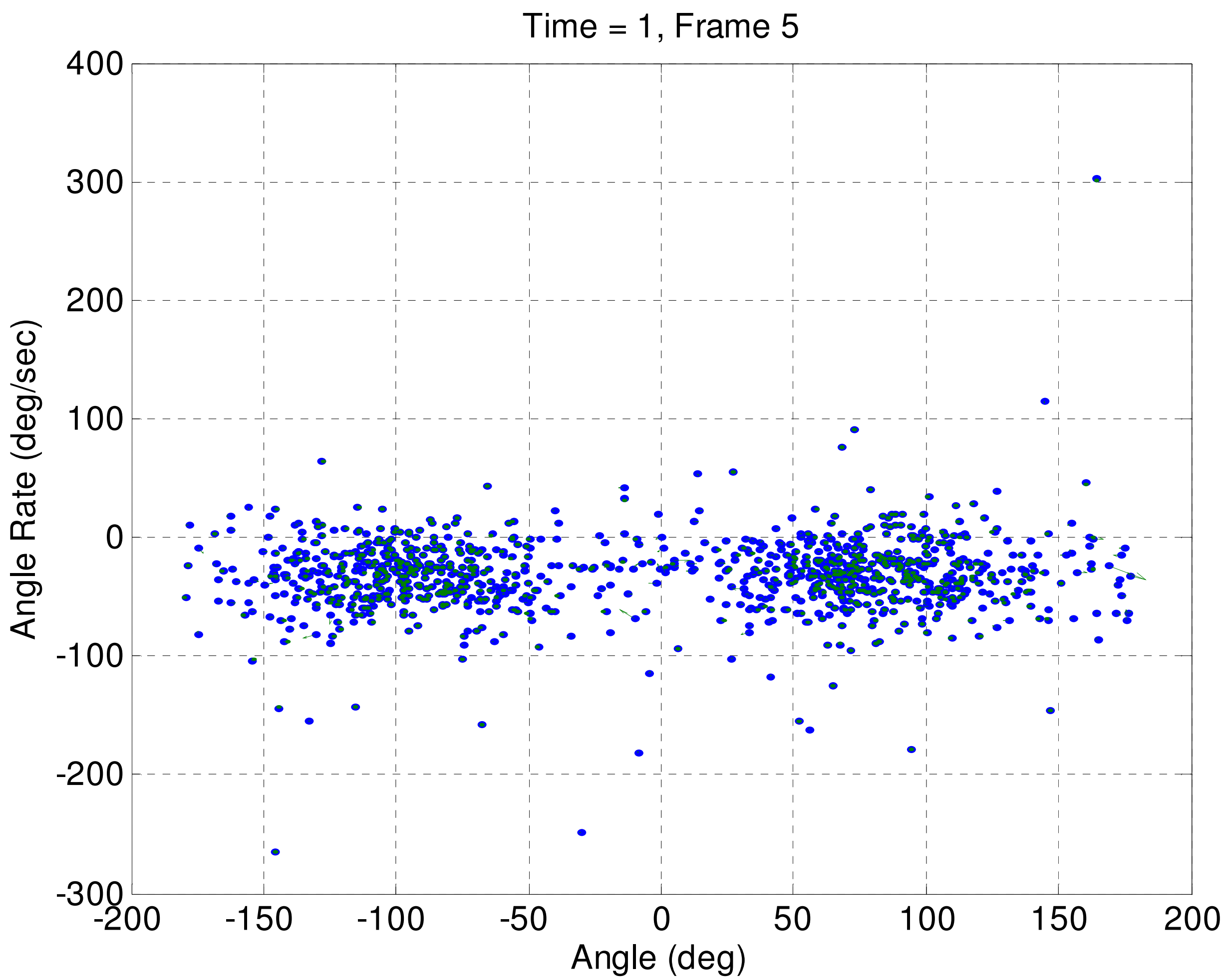


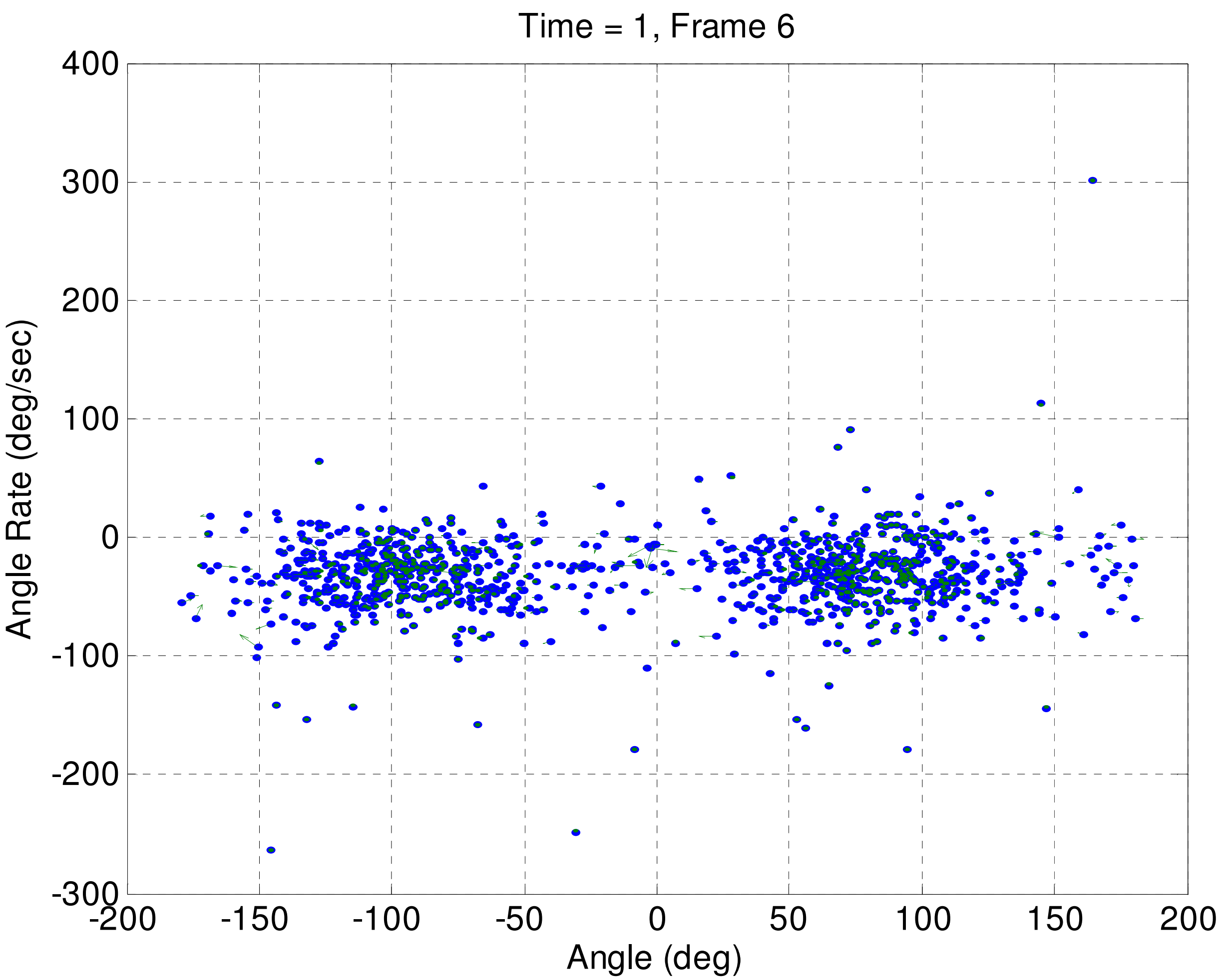
Time = 1, Frame 2
Angle Rate (deg/sec)
Angle (deg)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200

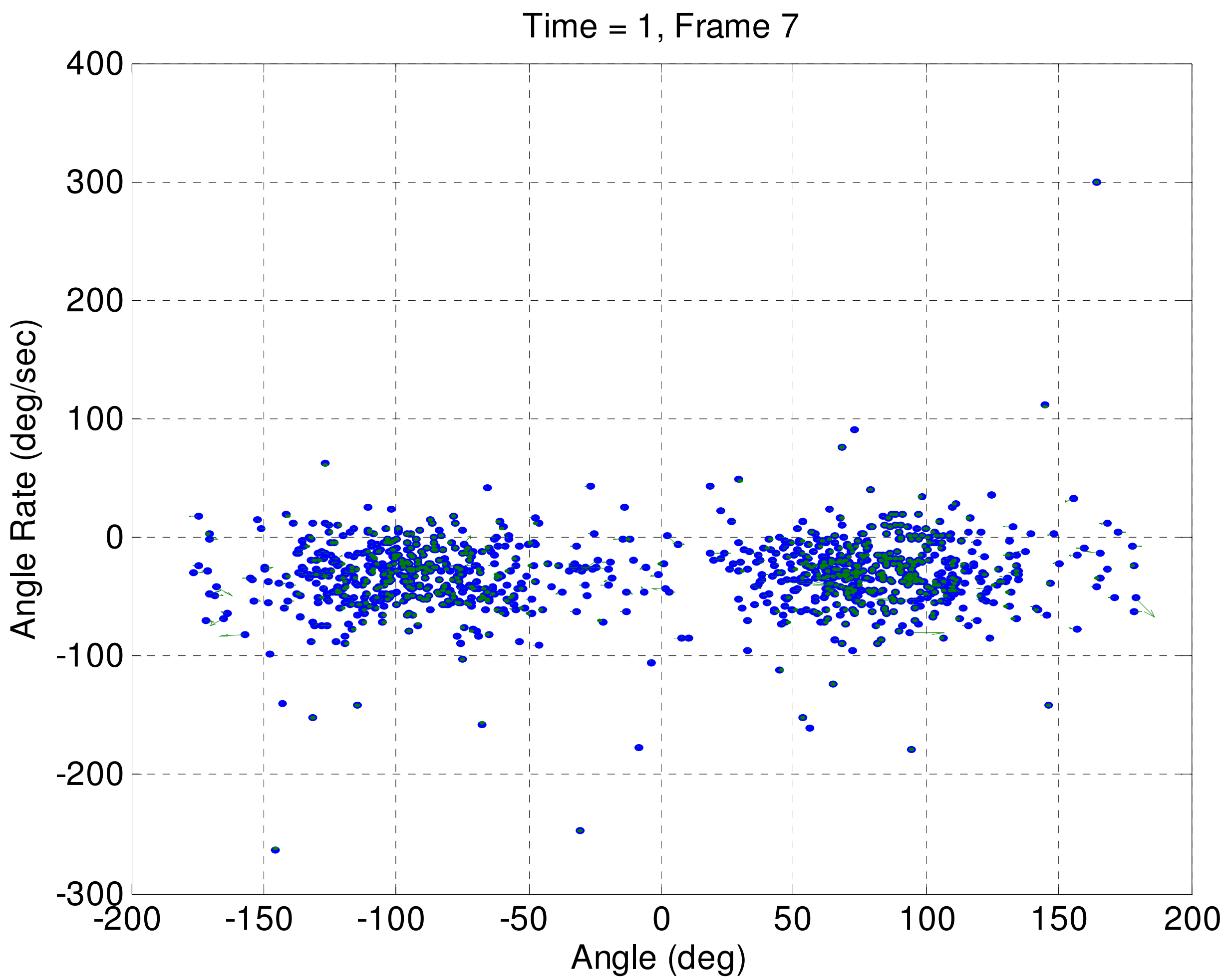


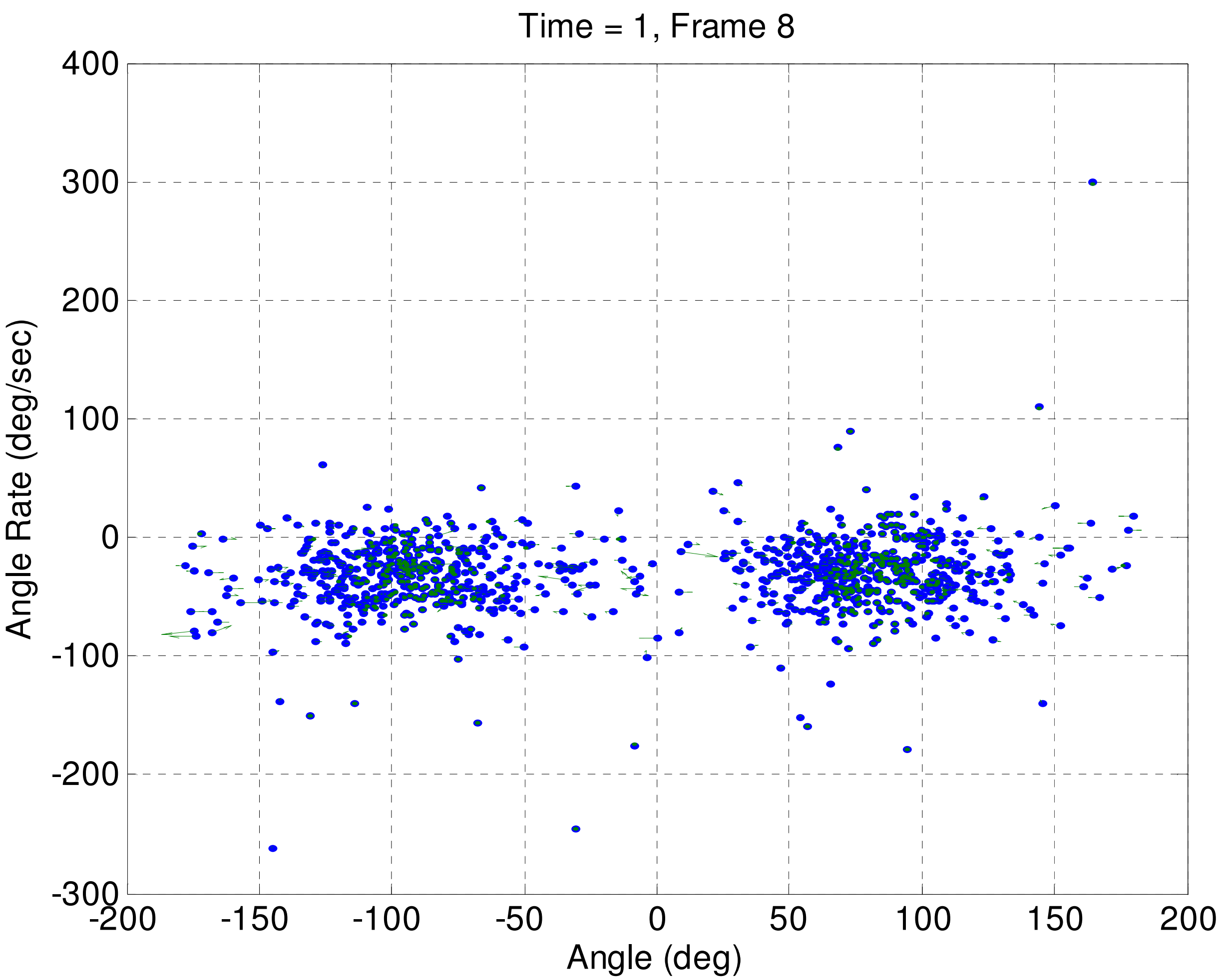
Time = 1, Frame 3
400
300
200
100
0
-100
-200
-300
Angle Rate (deg/sec)
-200
-150
-100
-50
0
50
100
150
200
Angle (deg)

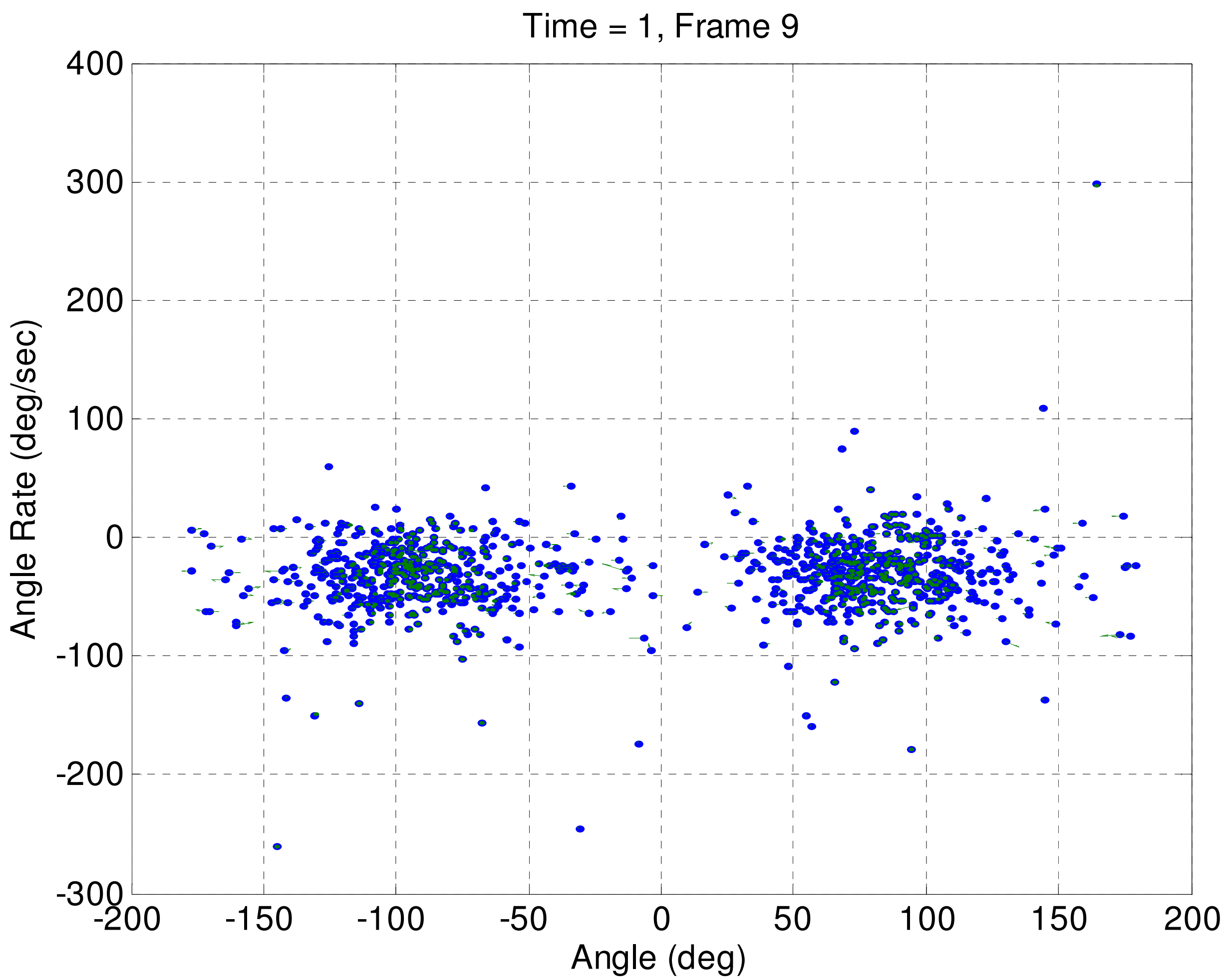

Time = 1, Frame 4
Angle Rate (deg/sec)
Angle (deg)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200

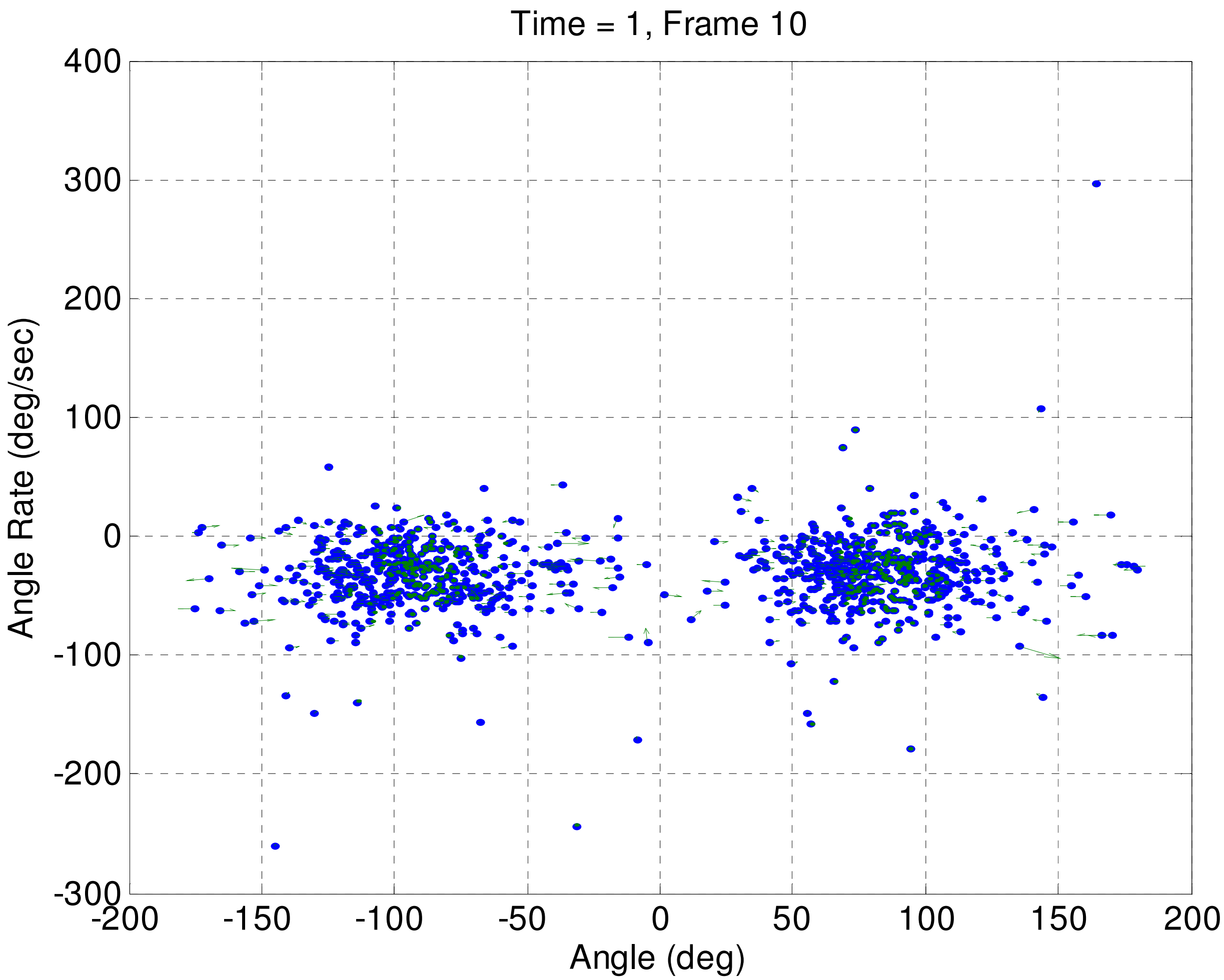

Time = 1, Frame 5
Angle Rate (deg/sec)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200
Angle (deg)


Time = 1, Frame 6
Angle Rate (deg/sec)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200
Angle (deg)


Time = 1, Frame 7
Angle Rate (deg/sec)
Angle (deg)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200


Time = 1, Frame 8
Angle Rate (deg/sec)
Angle (deg)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200


Time = 1, Frame 9
Angle Rate (deg/sec)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200
Angle (deg)



Time = 1, Frame 10
Angle Rate (deg/sec)
Angle (deg)
400
300
200
100
0
-100
-200
-300
-200
-150
-100
-50
0
50
100
150
200

direct integration of PDE

use the divergence form of the PDE :

$$div(q(x,\lambda)) = \eta(x,\lambda)$$

$$\eta = \frac{\partial q_1}{\partial x_1} + \frac{\partial q_2}{\partial x_2} + \ldots + \frac{\partial q_d}{\partial x_d}$$

$\tilde{q}$ = exact solution t o related problem $\mathrm{div}(\tilde{\mathrm{q}}) = \tilde{\eta}$

$\mathrm{div}(\mathrm{q} - \tilde{\mathrm{q}}) = \eta - \tilde{\eta}$

pick all but one component of $\mathrm{q} = \tilde{\mathrm{q}}$

$$\mathrm{q}_{\mathrm{j}} = \tilde{\mathrm{q}}_{\mathrm{j}} + \int^{x_j} \eta(x,\lambda) - \tilde{\eta}(x,\lambda)\, dx_j$$

$$q_j \approx \tilde{q}_j + x_j\left[\eta(x,\lambda) - \tilde{\eta}(x,\lambda)\right] + H.O.T.$$

most general solution for exact flow:

$$\frac{dx}{d\lambda} = f(x,\lambda)$$

$$\log(h) = -div(f) - \frac{\partial \log p}{\partial x} f$$

fundamental PDE for exact particle flow

the most general solution is:

$$\text{f} = \text{-C}^{\#} \log h + (I - C^{\#} C) y$$

in which C is a linear differential operator:

$$\text{C} = \frac{\partial \text{logp}}{\partial \text{x}} + div$$

$$C^{\#} = \text{generalized inverse of C}$$

$$\text{y} = \text{arbitrary d-dimensional vector}$$

could pick y to robustly stabilize the filter or random or other finite dimensional parametric approximation:

$$J = \left\| \log(h) + \frac{\partial \log p}{\partial x} f + div(f) \right\|^2 + r$$

let $\mathrm{f}(\mathrm{x}, \lambda) \approx \mathrm{A}(\lambda)\mathrm{x} + \mathrm{b}(\lambda)$

$$\mathrm{J} = \left\| \log(\mathrm{h}) + \frac{\partial \mathrm{logp}}{\partial \mathrm{x}} (Ax + b) + Tr(A) \right\|^2 + r$$

(1) solve for A & b to minimize J at each particle x

(2) could let $\mathrm{A} = -\mathrm{BB}^{\mathrm{T}}$ to force stability of flow

(3) could also add penalty to make flow robustly stable

(4) note that $\mathrm{div(f)} = \mathrm{Tr(A)} = \sum \lambda_{\mathrm{j}}(A)$

(5) however, computational complexity is d^6, but for sparse A this can be reduced to only d^3

hybrid method:

We want to find a flow $f(x,\lambda)$ that satisfies the following PDE:

$$\frac{\partial \log p(x,\lambda)}{\partial x} f + div(f) = -\log h(x)$$

Without loss of generality, let $f(x,\lambda) = A(\lambda)x + b(\lambda) + c(x,\lambda)$ in which A & b are computed from our exact solution with g & h Gaussian using an EKF (as usual for our exact flow):

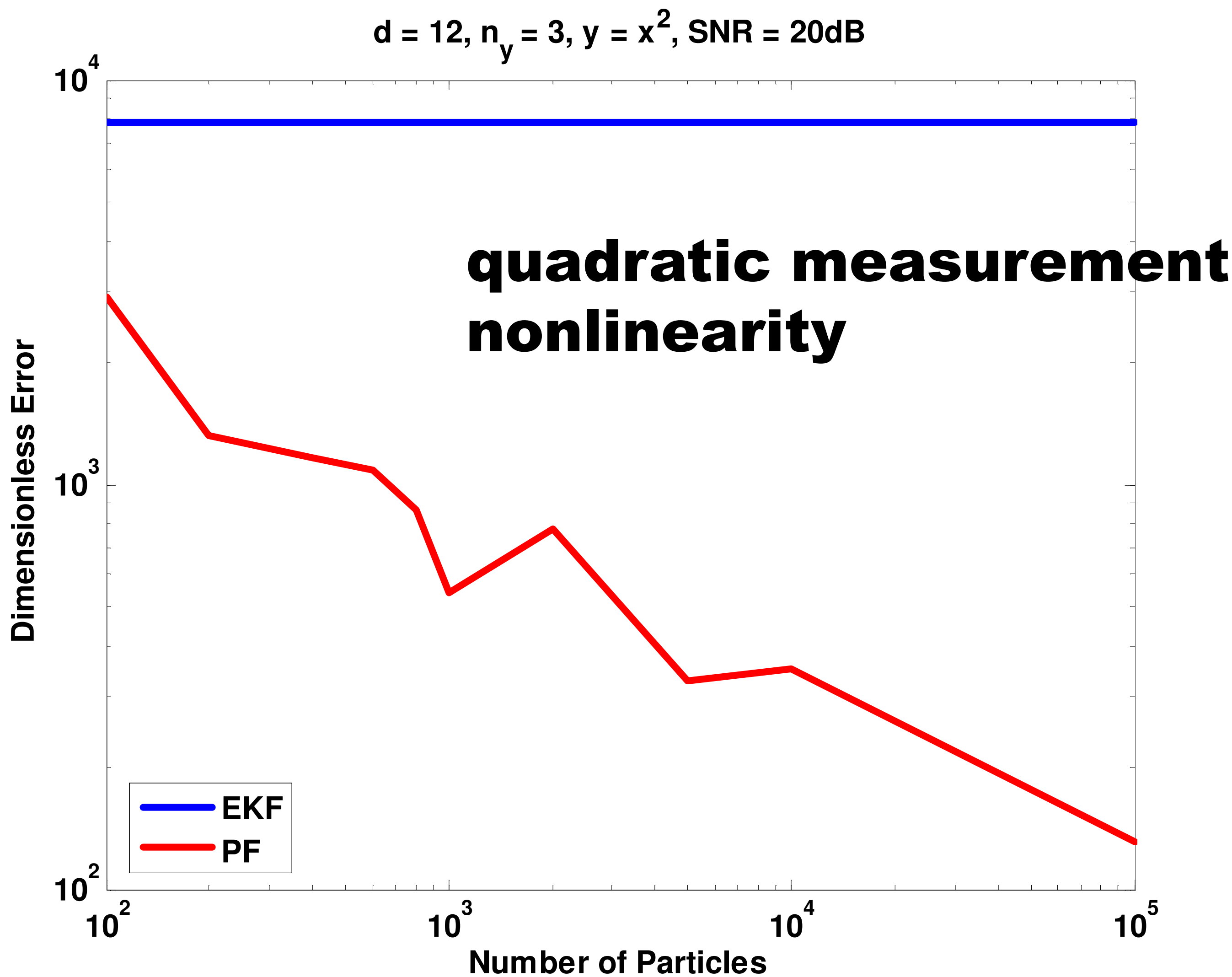
$$\frac{\partial \log p}{\partial x}(Ax+b+c)+Tr(A)+div(c) = -\log(h)$$

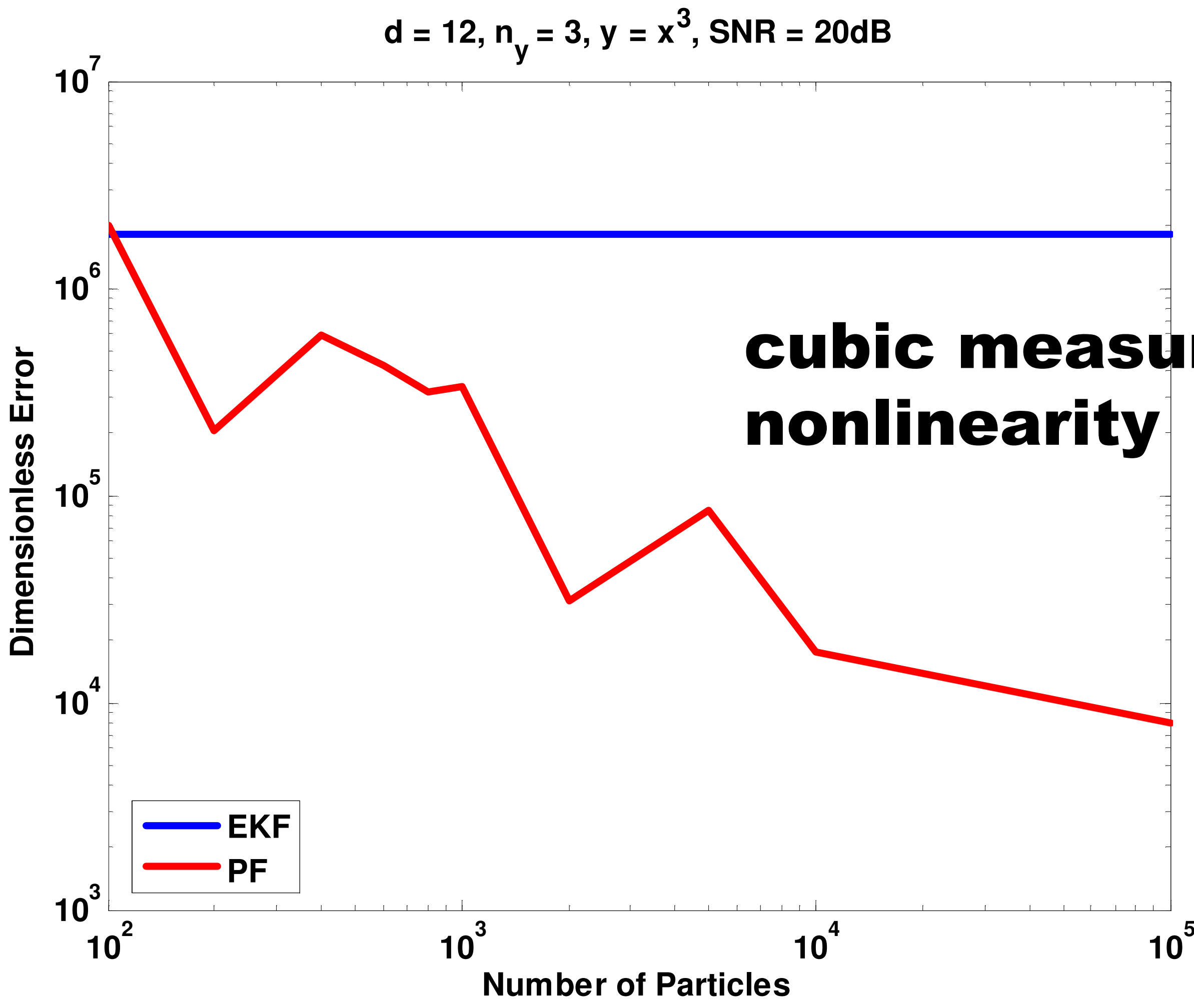
Assume that $div(c) \approx 0$, and compute the minimum norm $c(x,\lambda)$:

$$c \approx -\left(\frac{\partial \log p}{\partial x}\right)^{\#}\left[\frac{\partial \log p}{\partial x}(Ax+b)+Tr(A)+\log(h)\right]$$

in which $(.)^{\#}$ denotes the pseudo-inverse of $(.)$:

$$\left(\frac{\partial \log p}{\partial x}\right)^{\#} = \left(\frac{\partial \log p}{\partial x}\right)^{T} / \left\|\frac{\partial \log p}{\partial x}\right\|^{2}$$


d = 12, n_y = 3, y = x^2, SNR = 20dB
quadratic measurement nonlinearity
Dimensionless Error
Number of Particles
EKF
PF
10^4
10^3
10^2
10^2
10^3
10^4
10^5

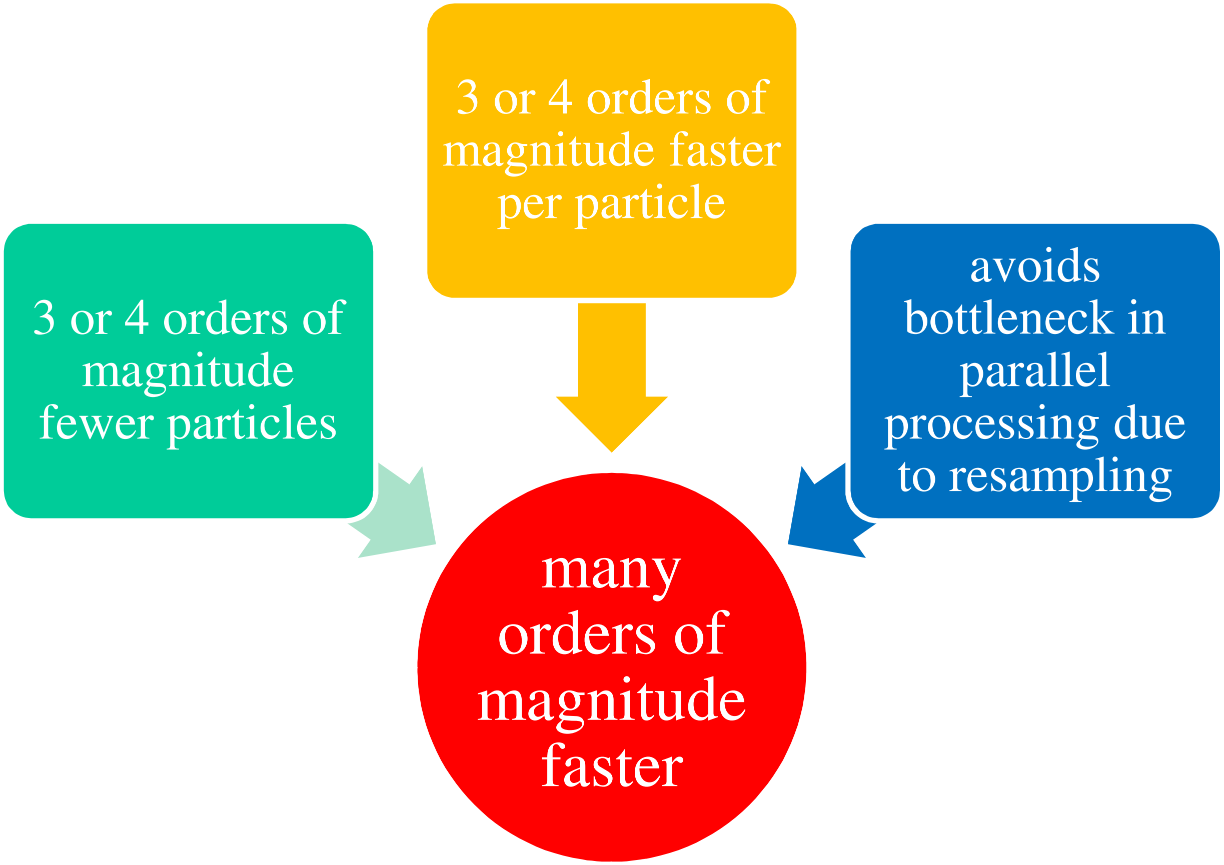


cubic measurement nonlinearity

particle flow filter

- orders of magnitude faster than standard particle filters
- orders of magnitude more accurate than the extended Kalman filter for difficult nonlinear problems
- solves particle degeneracy problem using particle flow induced by log-homotopy for Bayes' rule
- no resampling of particles
- no proposal density
- no importance sampling & no MCMC methods
- unnormalized log probability density
- embarrassingly parallelizable w/o resampling bottleneck (unlike other particle filters)
- exploits smoothness & regularity of densities

particle flow filter is many orders of magnitude faster real time computation (for the same or better estimation accuracy)



BACKUP

exact particle flow & Poisson's equation:

$$\frac{dx}{d\lambda} = f(x,\lambda)$$

$$q(x,\lambda) = p(x,\lambda) f(x,\lambda)$$

$$\log h(x) p(x,\lambda) = -Tr\left[\frac{\partial q(x,\lambda)}{\partial x}\right] = -div(q)$$

divergence form of PDE

obviously there is no unique solution,

so pick the unique minimum norm solution:

$$\mathrm{q}(\mathrm{x},\lambda) = \frac{\partial \mathrm{V}(\mathrm{x},\lambda)}{\partial \mathrm{x}}$$

$$Tr\left[\frac{\partial^2 V(x,\lambda)}{\partial x^2}\right] = -\log h(x) p(x,\lambda)$$

Poisson's equation

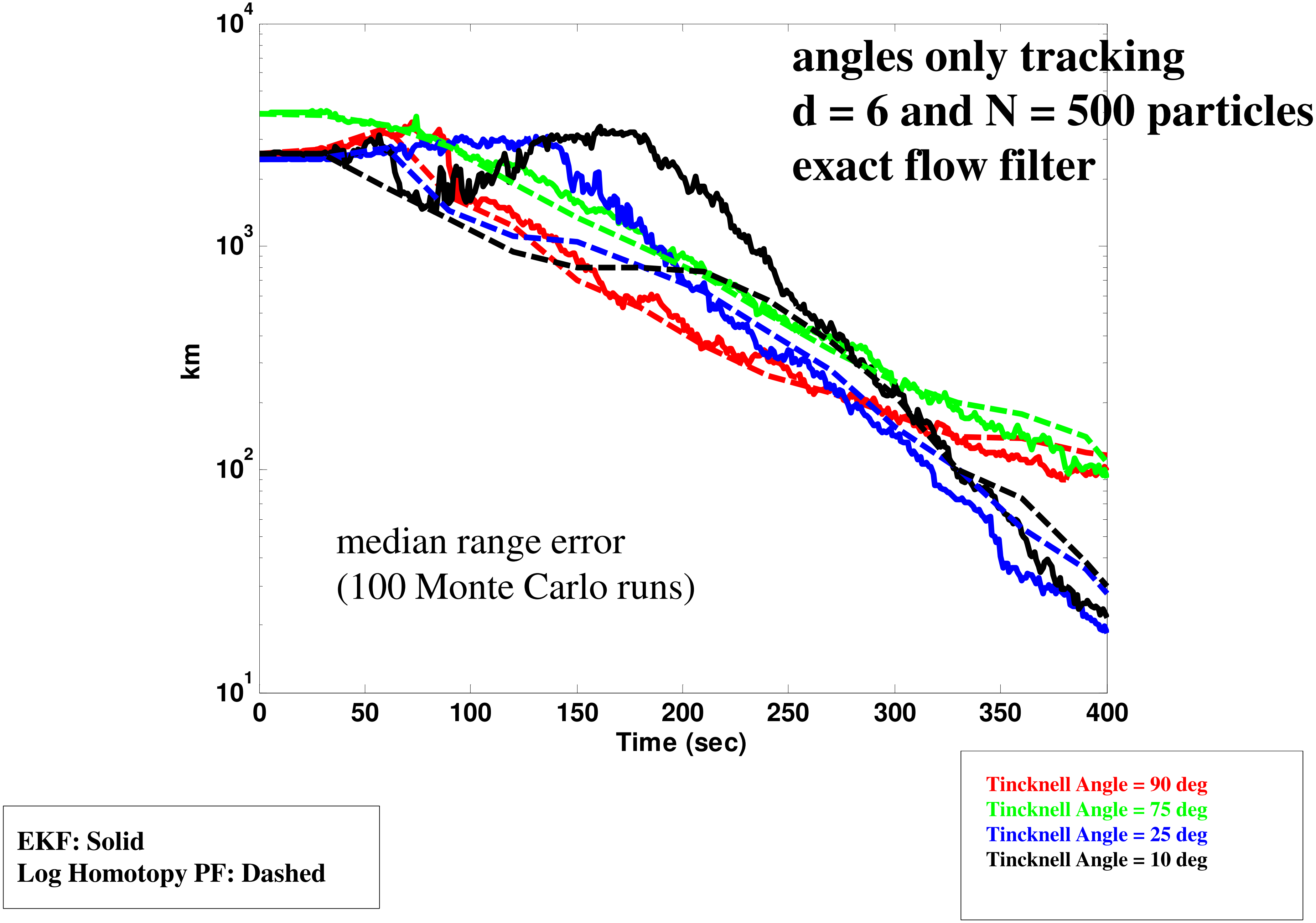
$$\frac{dx}{d\lambda} = f(x,\lambda) = \frac{\partial V(x,\lambda)}{\partial x} / p(x,\lambda)$$

incompressible particle flow

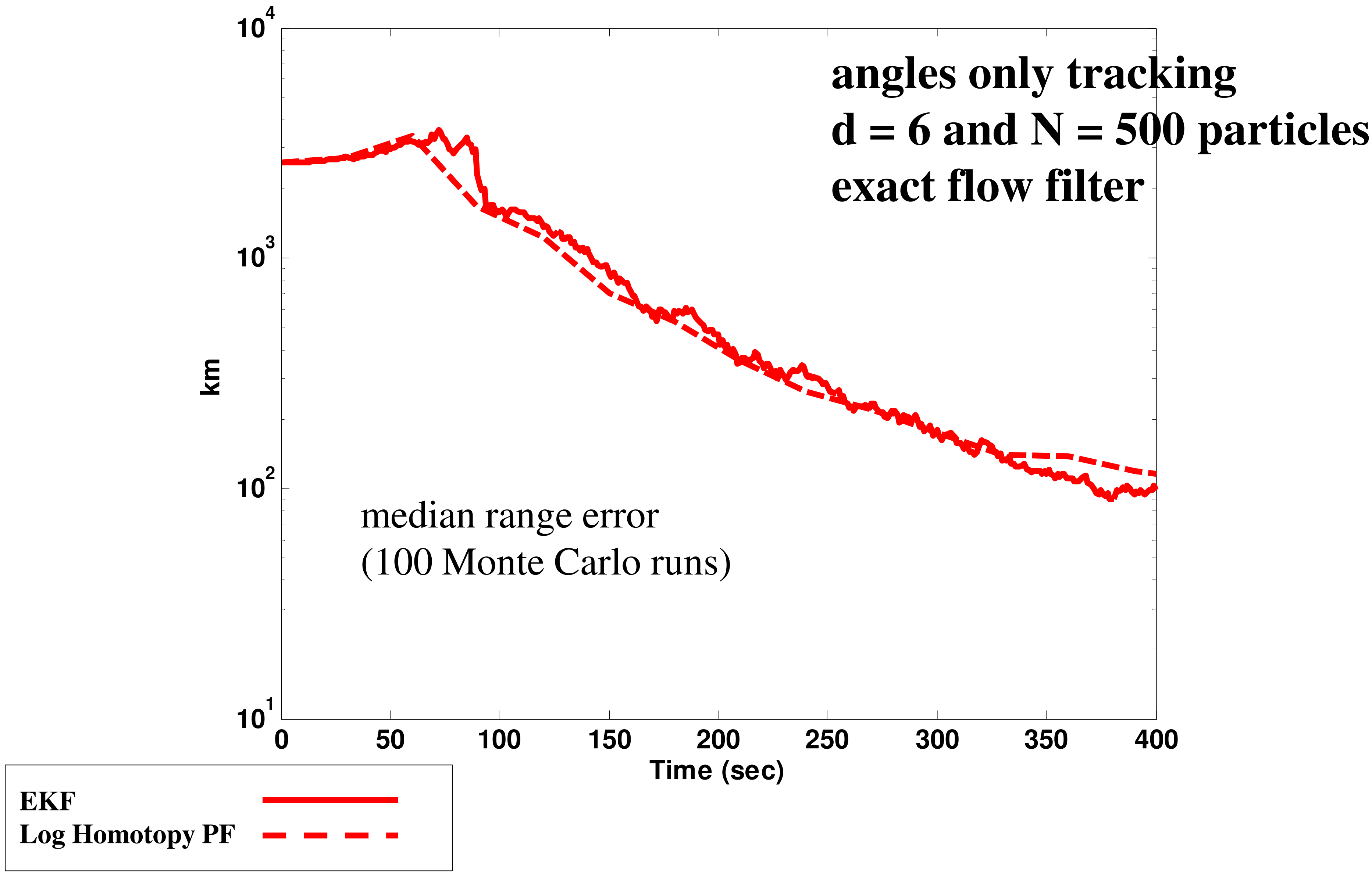
$$\frac{\mathrm{dx}}{\mathrm{d}\lambda} = {-\log(h)\left(\frac{\partial \log p}{\partial x}\right)^T} \Big/ {\left\|\frac{\partial \log p}{\partial x}\right\|^2}$$

$$\frac{dx}{d\lambda} = 0 \qquad \text{for zero gradient}$$

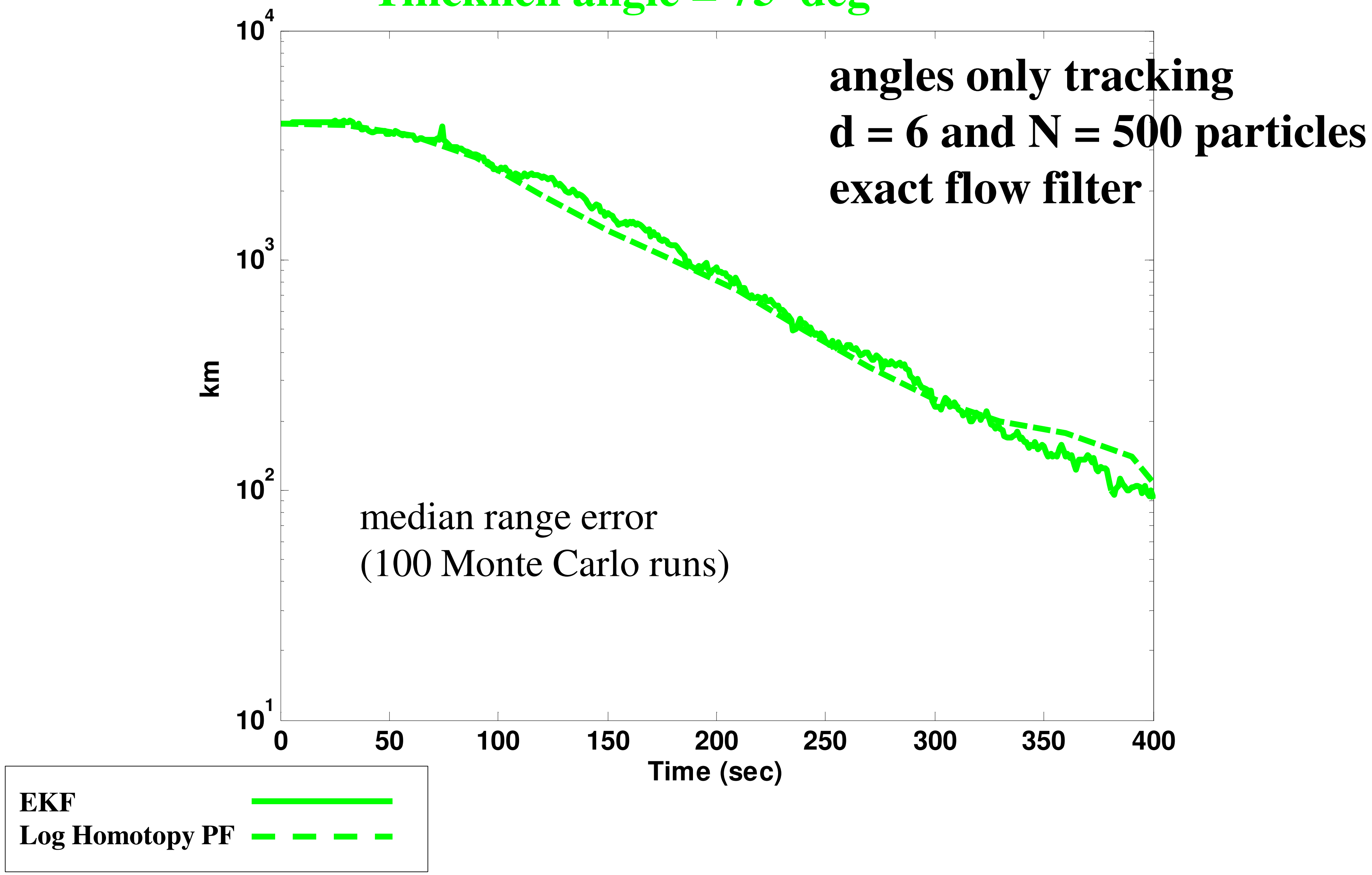
method to solve PDE	how to pick unique solution	computation
1. generalized inverse of linear differential operator	minimum norm*	fast Poisson solver in d-dimensions or Coulomb's law or other
2. Poisson's equation	gradient of potential* (assume irrotational flow)	fast Poisson solver in d-dimensions or Coulomb's law or other
3. generalized inverse of gradient of log-homotopy	assume incompressible flow (i.e., divergence free flow)	fast (but need to compute the gradient from random points)
4. most general solution	most robustly stable filter or random pick, etc.	fast (but need to compute the gradient from random points)
5. separation of variables (Gaussian)	pick solution of specific form (polynomial)	extremely fast (formula)
6. separation of variables (exponential family)	pick solution of specific form (finite basis functions)	very fast (formula)
7. variational formulation (Gauss & Hertz)	convex function minimization	ODEs
8. optimal control formulation	convex functional minimization	Euler-Lagrange PDEs (or maybe ODES for nice problem)
9. direct integration (of first order linear PDE in divergence form)	choice of d-1 arbitrary functions	one-dimensional integral
10. generalized method of characteristics	more conditions (e.g., curl = given & chain rule)	ODEs from chain rule
11. another homotopy (inspired by Gromov's h-principle)	initial condition of ODE & uniqueness of sol. to ODE	ODEs from homotopy
12. finite dimensional parametric flow (e.g., f = Ax+b)	non-singular matrix to invert	d^3 or d^6 (least squares for d or d^2 parameters, i.e., A & b)
13. Fourier transform of PDE (divergence form of PDE has constant coefficients)	minimum norm* or most stable flow	Gaussian sum makes inverse very fast (by inspection)

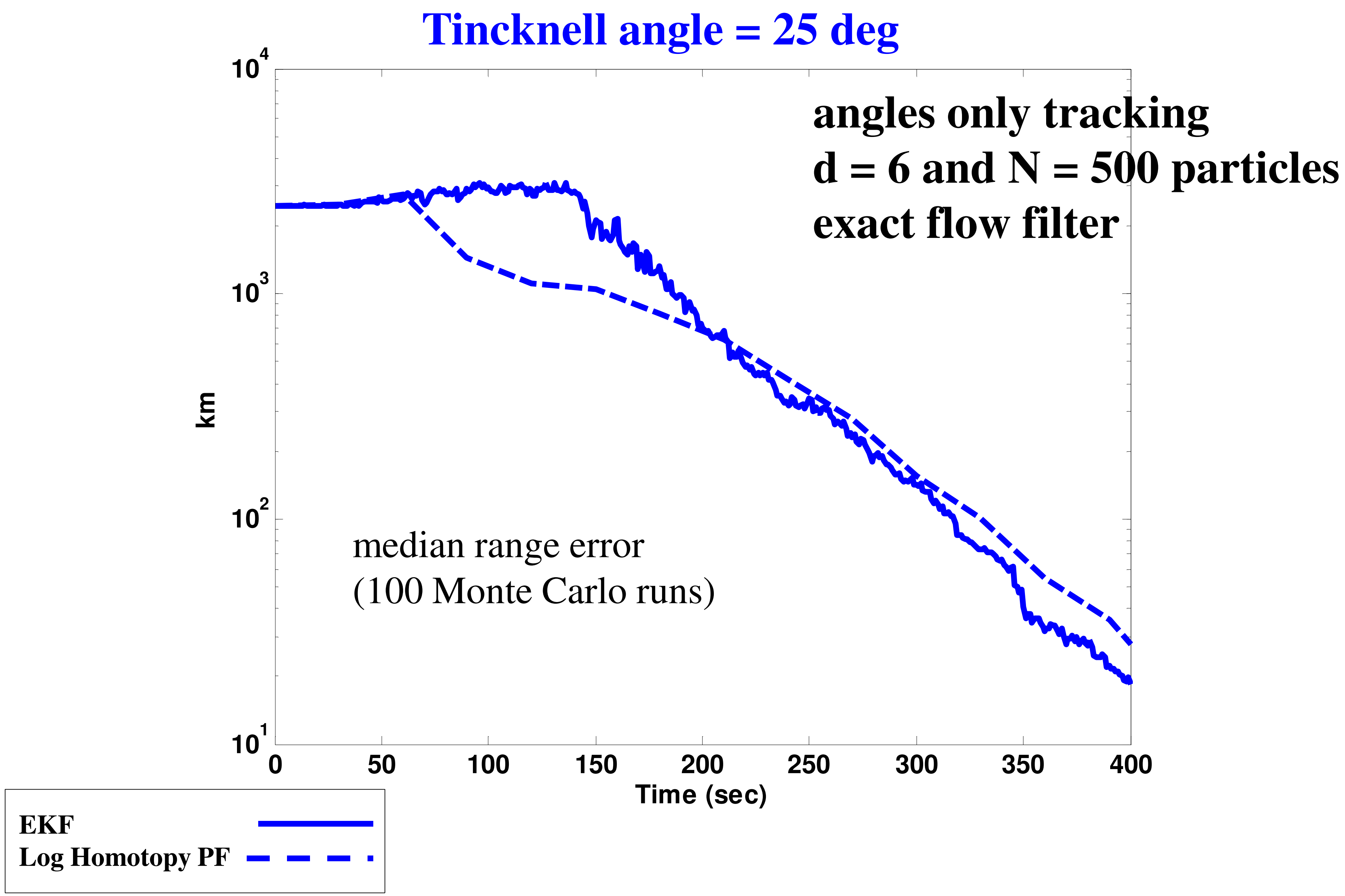

angles only tracking
d = 6 and N = 500 particles
exact flow filter
median range error
(100 Monte Carlo runs)
km
Time (sec)
10^4
10^3
10^2
10^1
0
50
100
150
200
250
300
350
400
Tincknell Angle = 90 deg
Tincknell Angle = 75 deg
Tincknell Angle = 25 deg
Tincknell Angle = 10 deg
EKF: Solid
Log Homotopy PF: Dashed

Tincknell angle = 90 deg

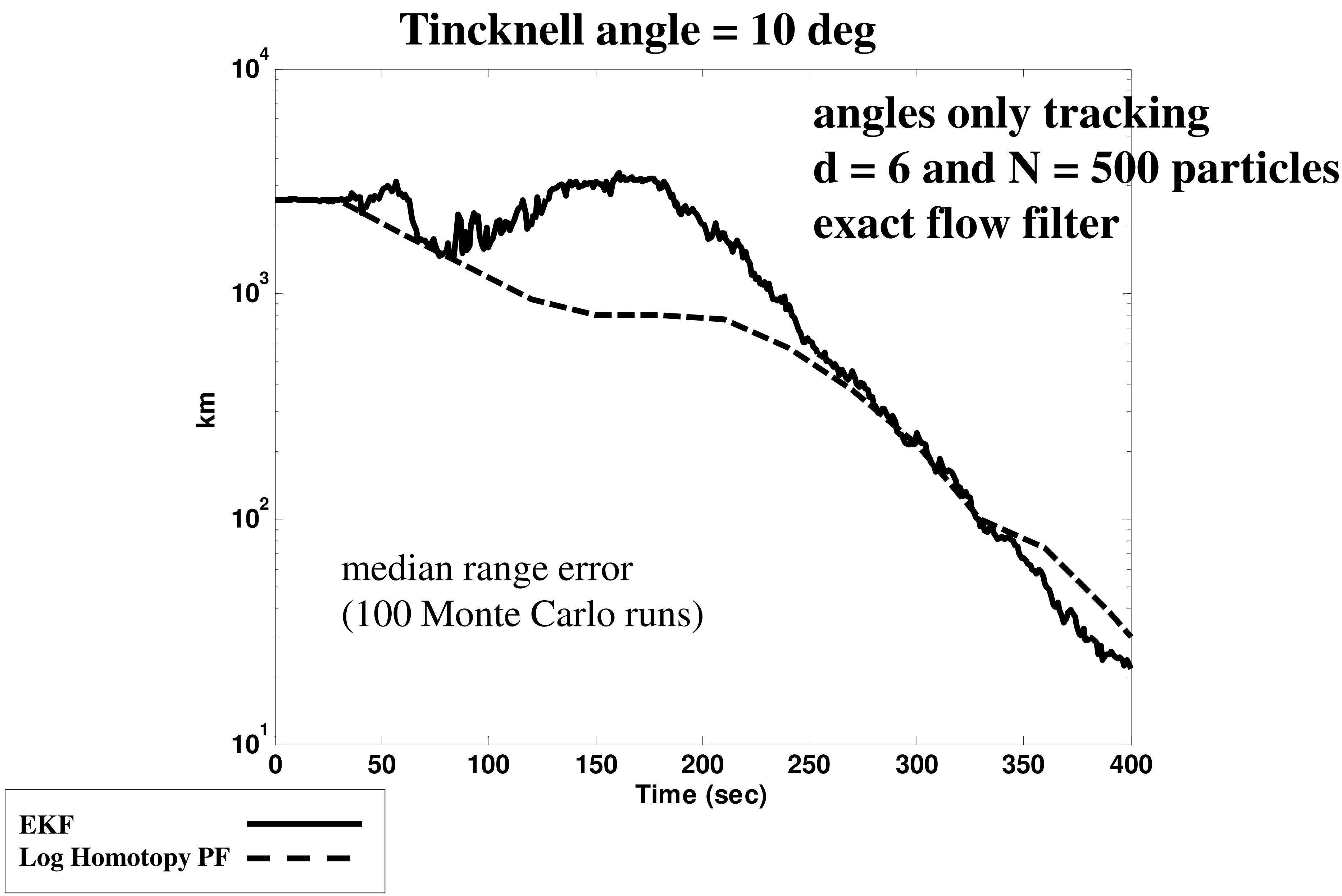


Tincknell angle = 75 deg





Tincknell angle = 25 deg
angles only tracking
d = 6 and N = 500 particles
exact flow filter
median range error
(100 Monte Carlo runs)
km
Time (sec)
EKF
Log Homotopy PF


Tincknell angle = 10 deg
angles only tracking
d = 6 and N = 500 particles
exact flow filter
median range error
(100 Monte Carlo runs)
km
Time (sec)
EKF
Log Homotopy PF

exact particle flow for Gaussian densities:

$$\frac{dx}{d\lambda} = f(x, \lambda)$$

$$\log(h) = -div(f) - \frac{\partial \log p}{\partial x} f$$

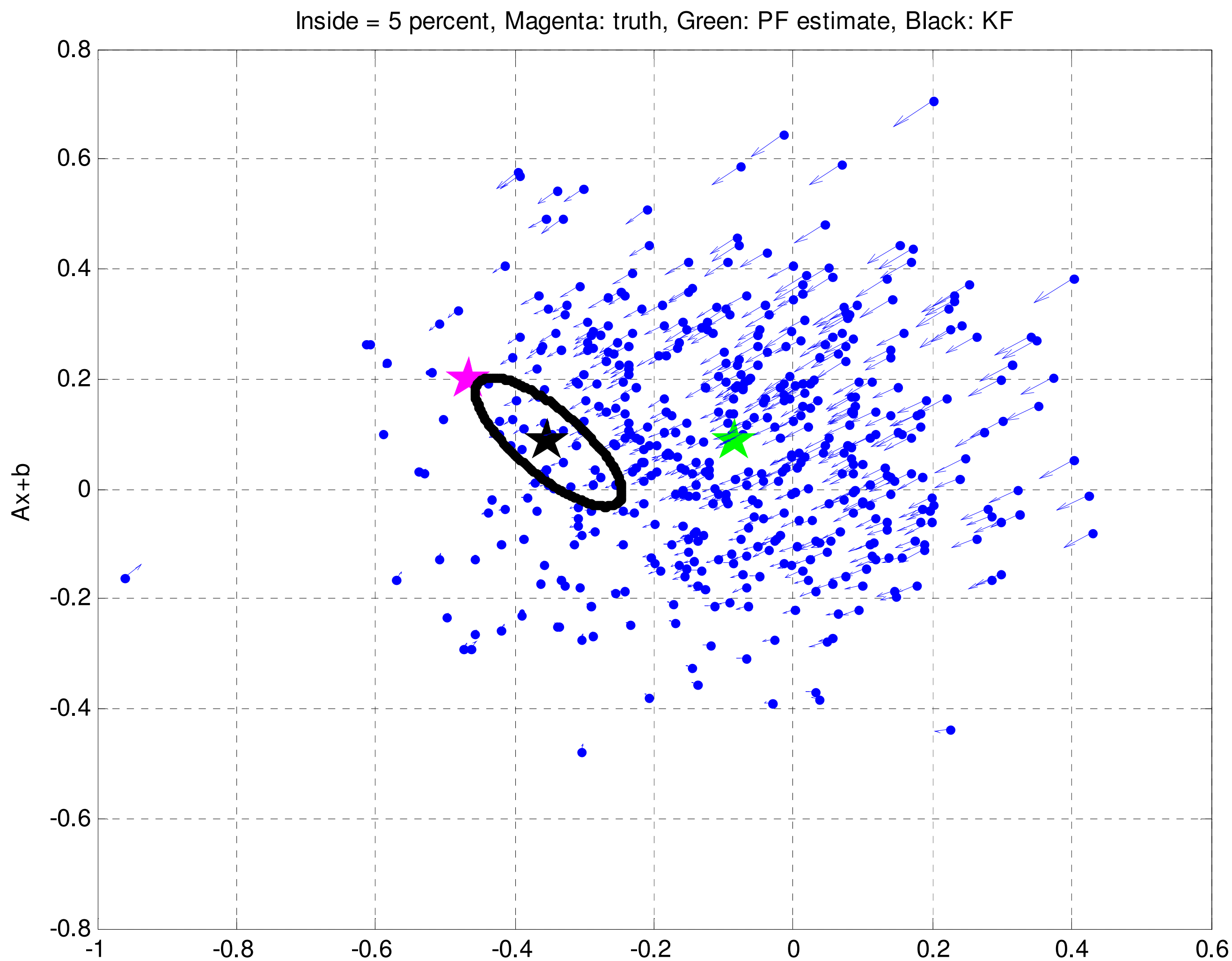
for g & h Gaussian, we can solve for f exactly:

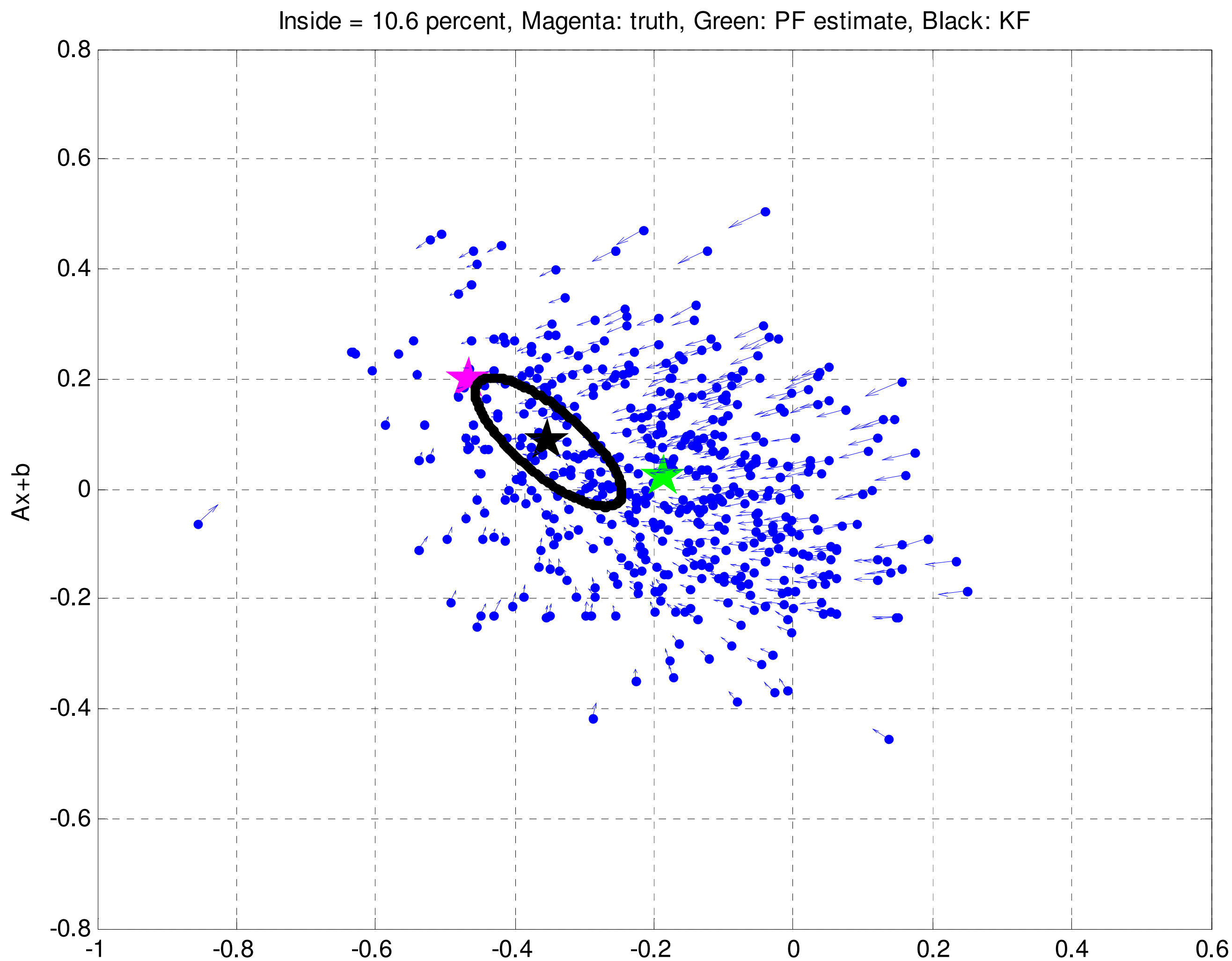
$$f = Ax + b$$

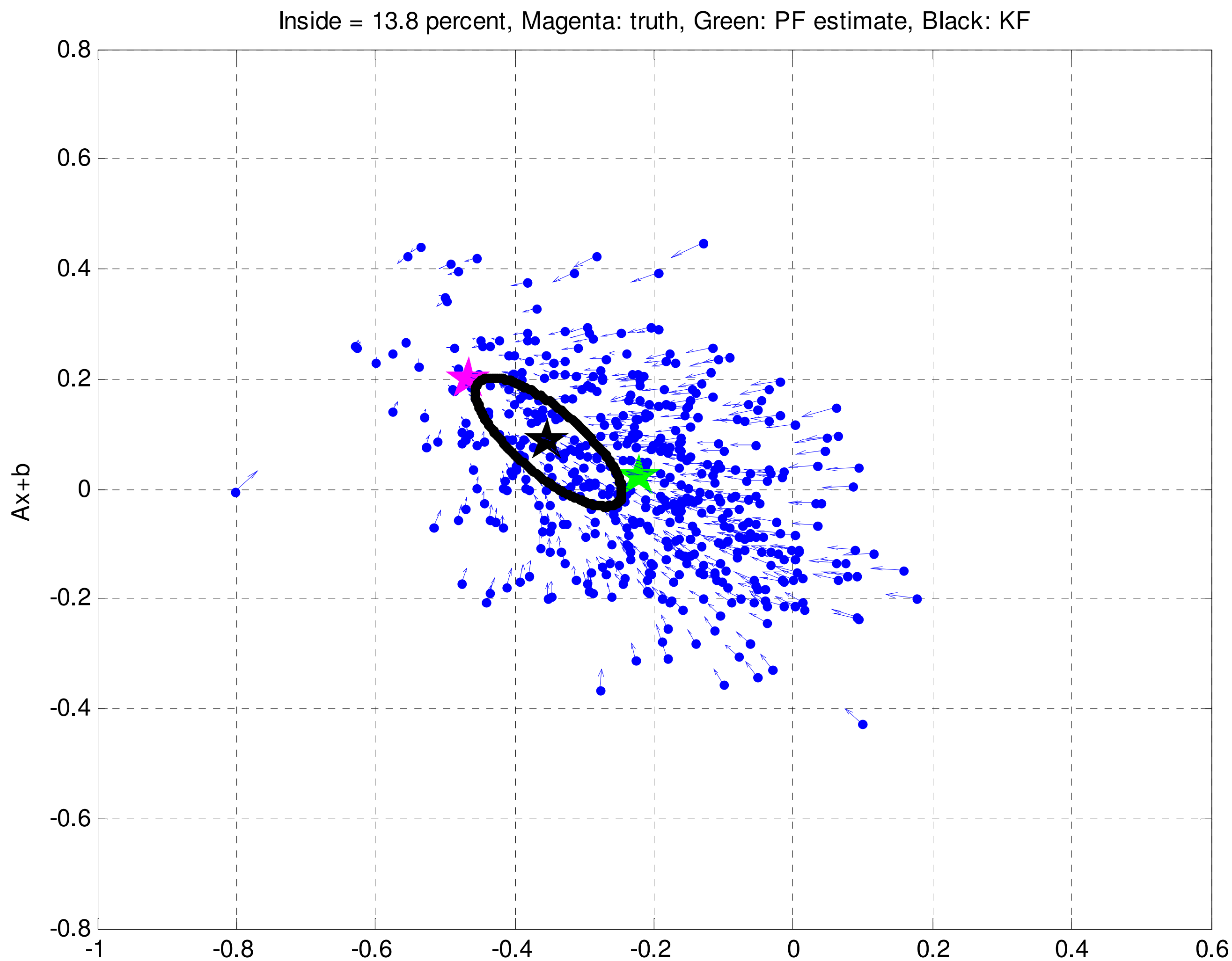
$$A = -\frac{1}{2} PH^T \left[\lambda HPH^T + R\right]^{-1} H$$

$$b = (I + 2\lambda A)\left[(I + \lambda A)PH^T R^{-1} z + A\bar{x}\right]$$

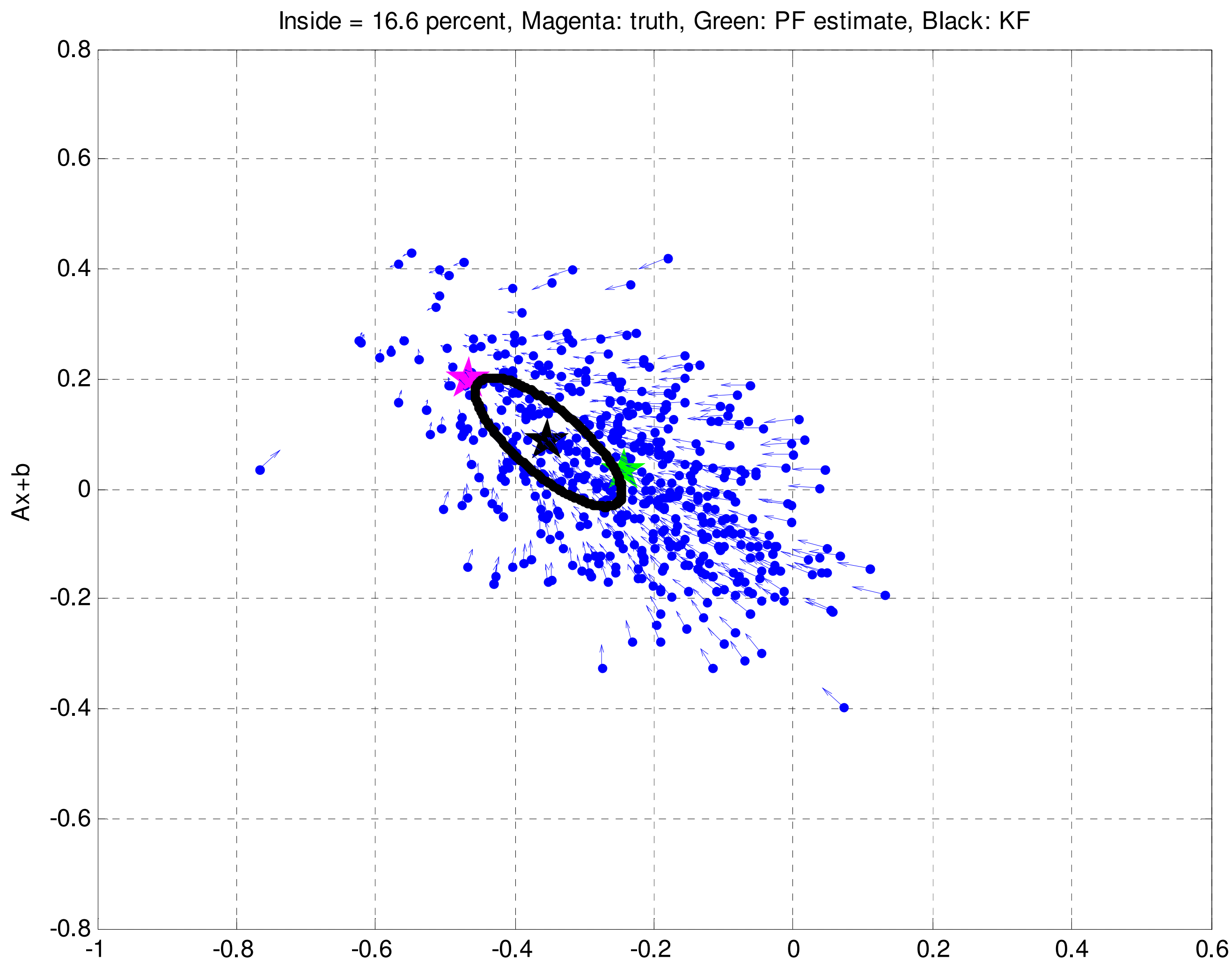
automatically stable under very mild conditions & extremely fast

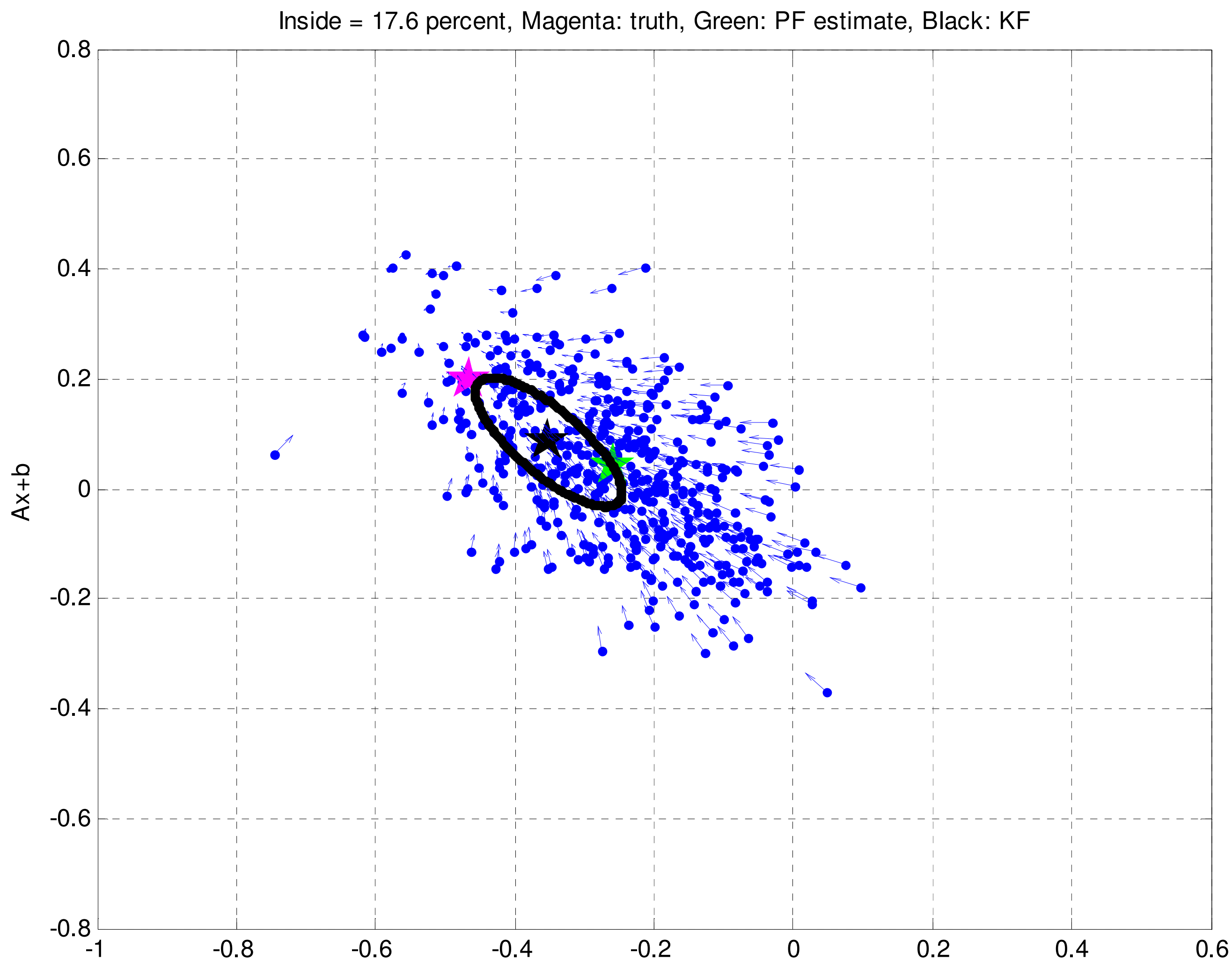

Inside = 5 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

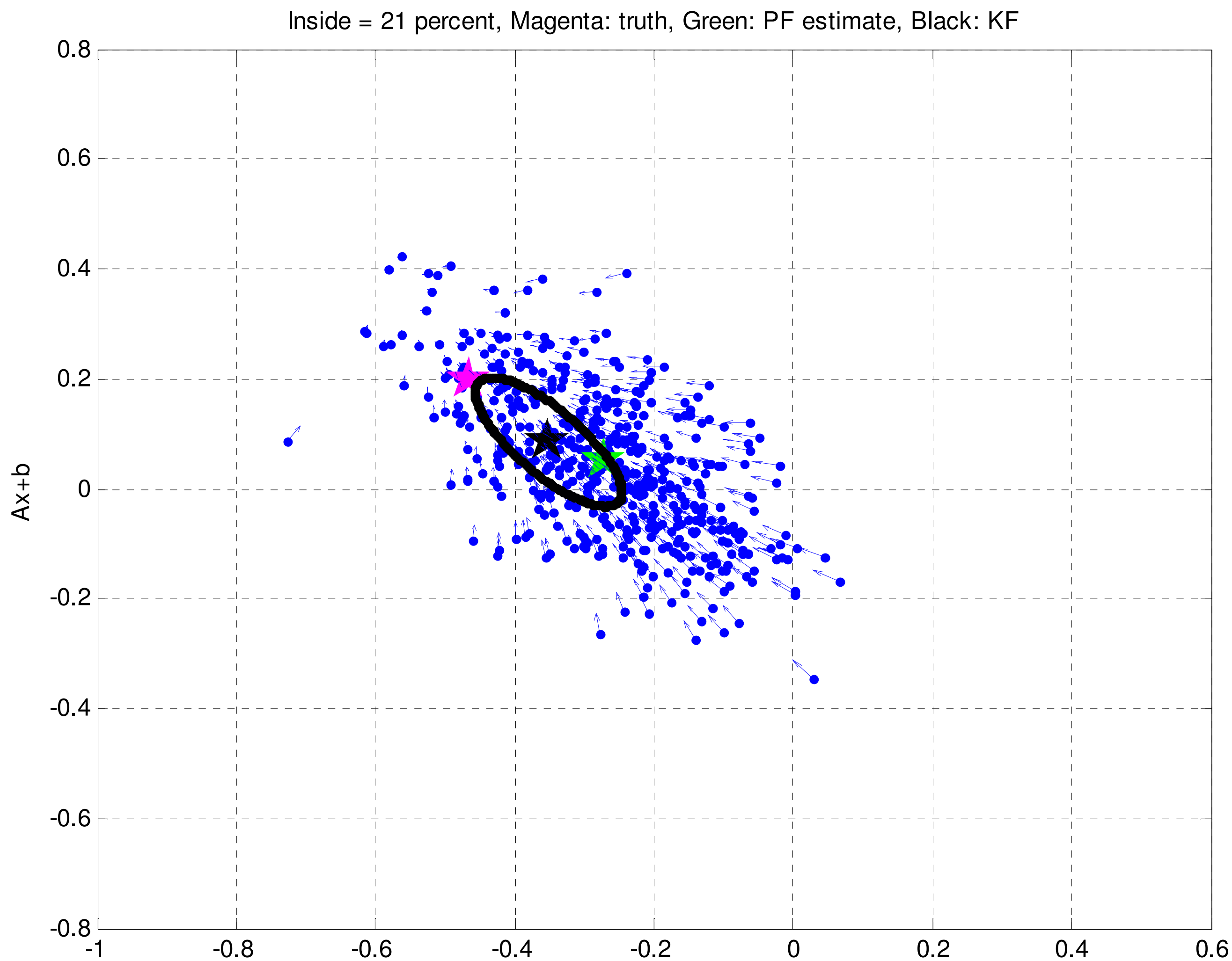

Inside = 10.6 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

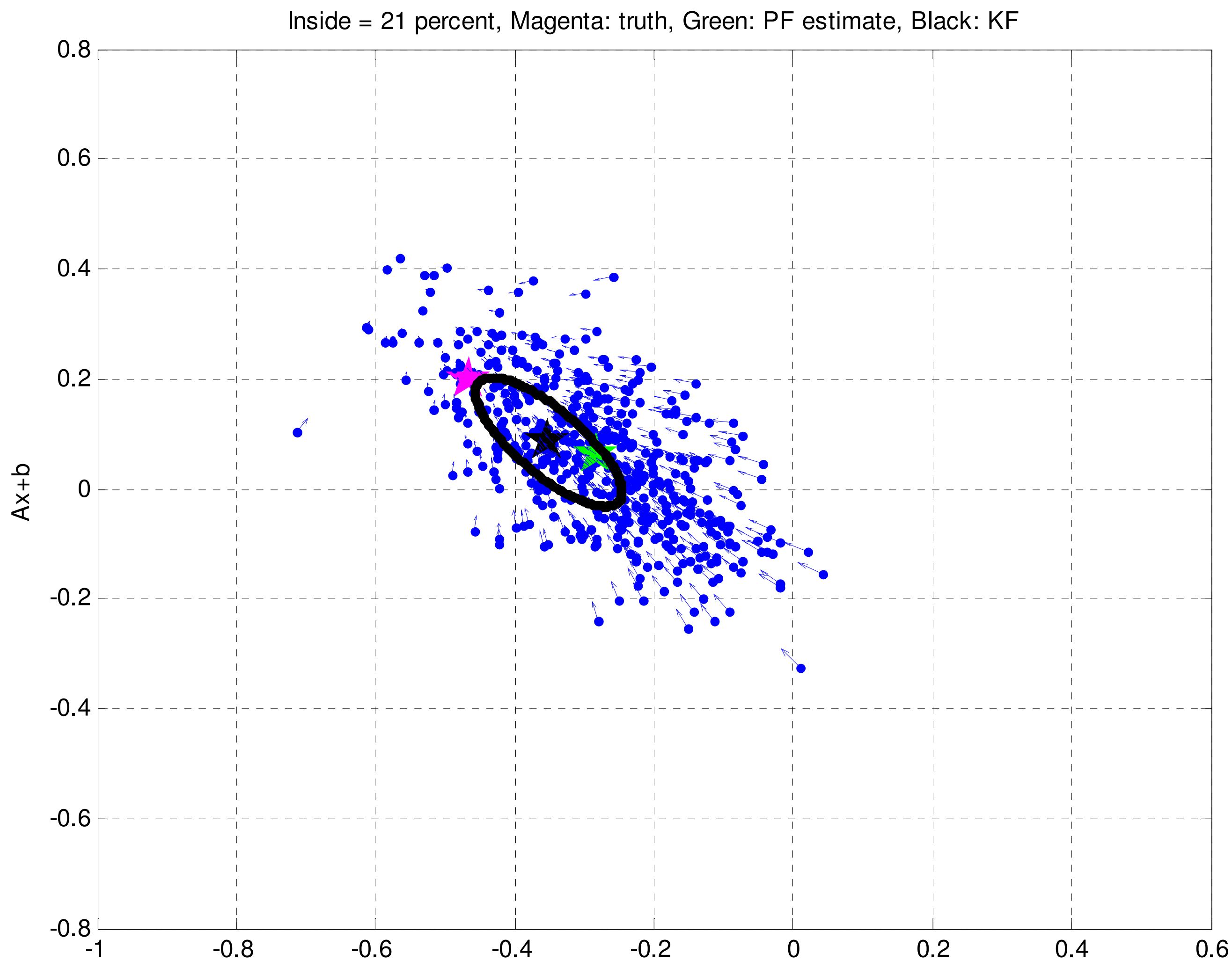


Inside = 13.8 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

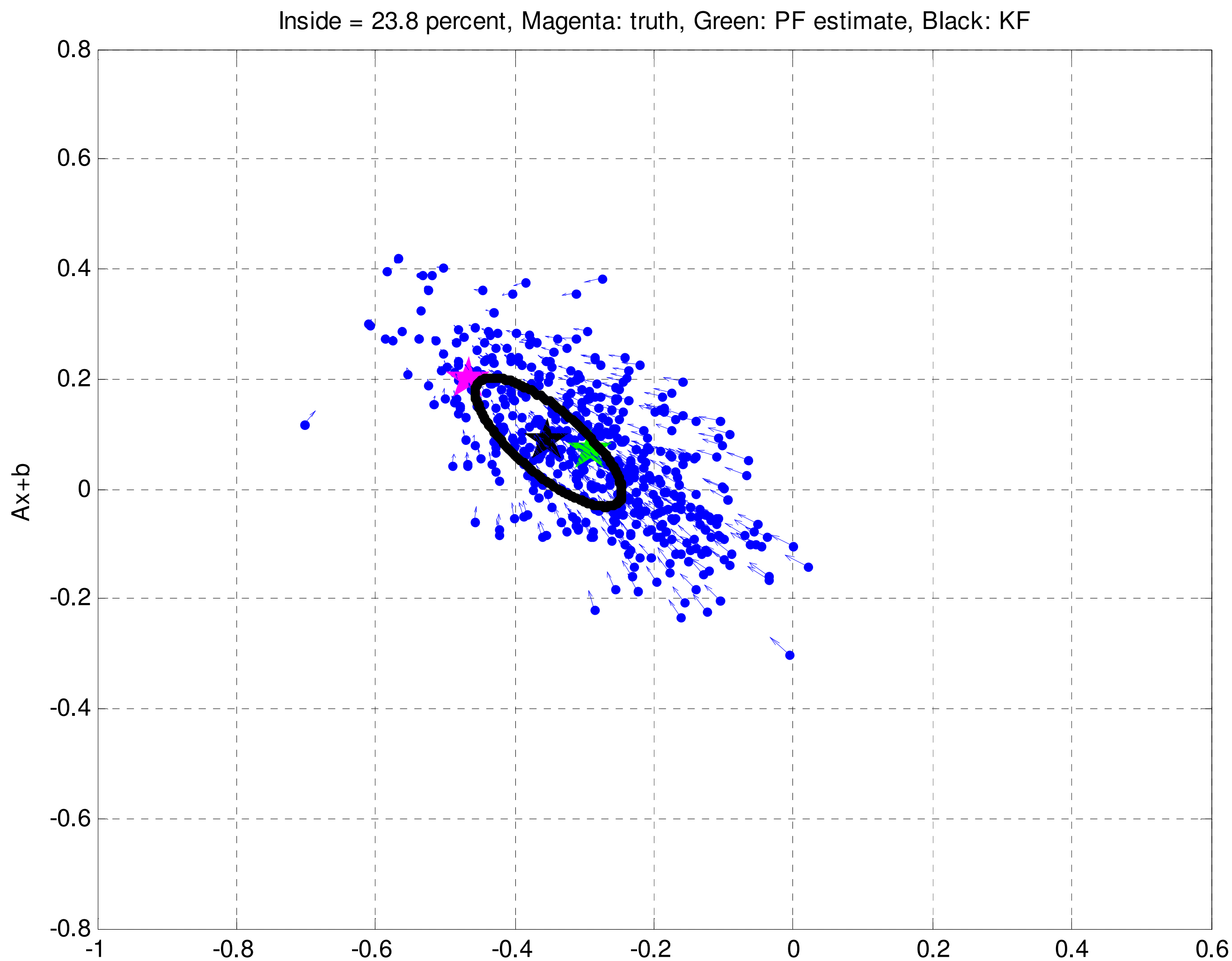

Inside = 16.6 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6


Inside = 17.6 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6


Inside = 21 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6



Inside = 21 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6



Inside = 23.8 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

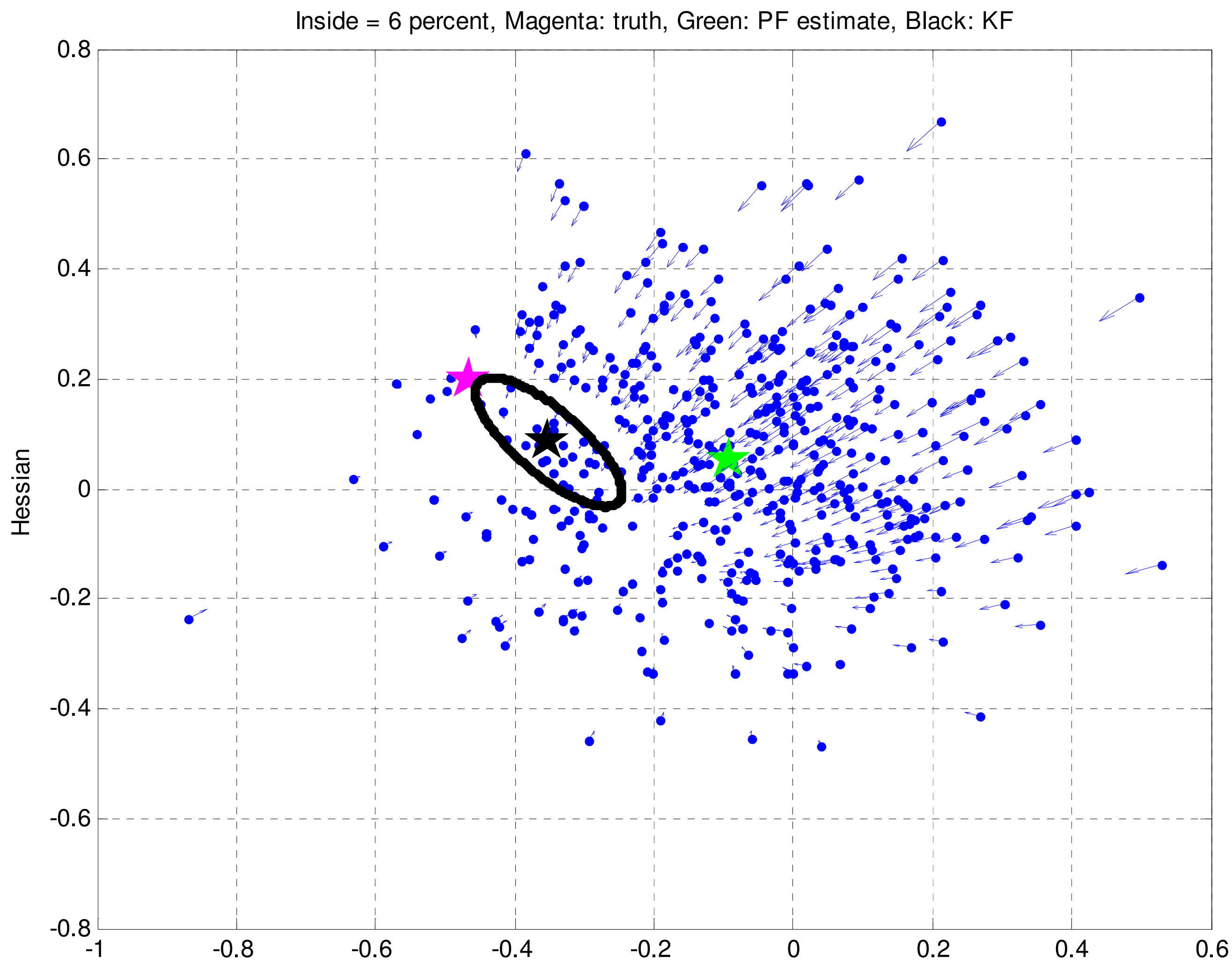

Inside = 26.4 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

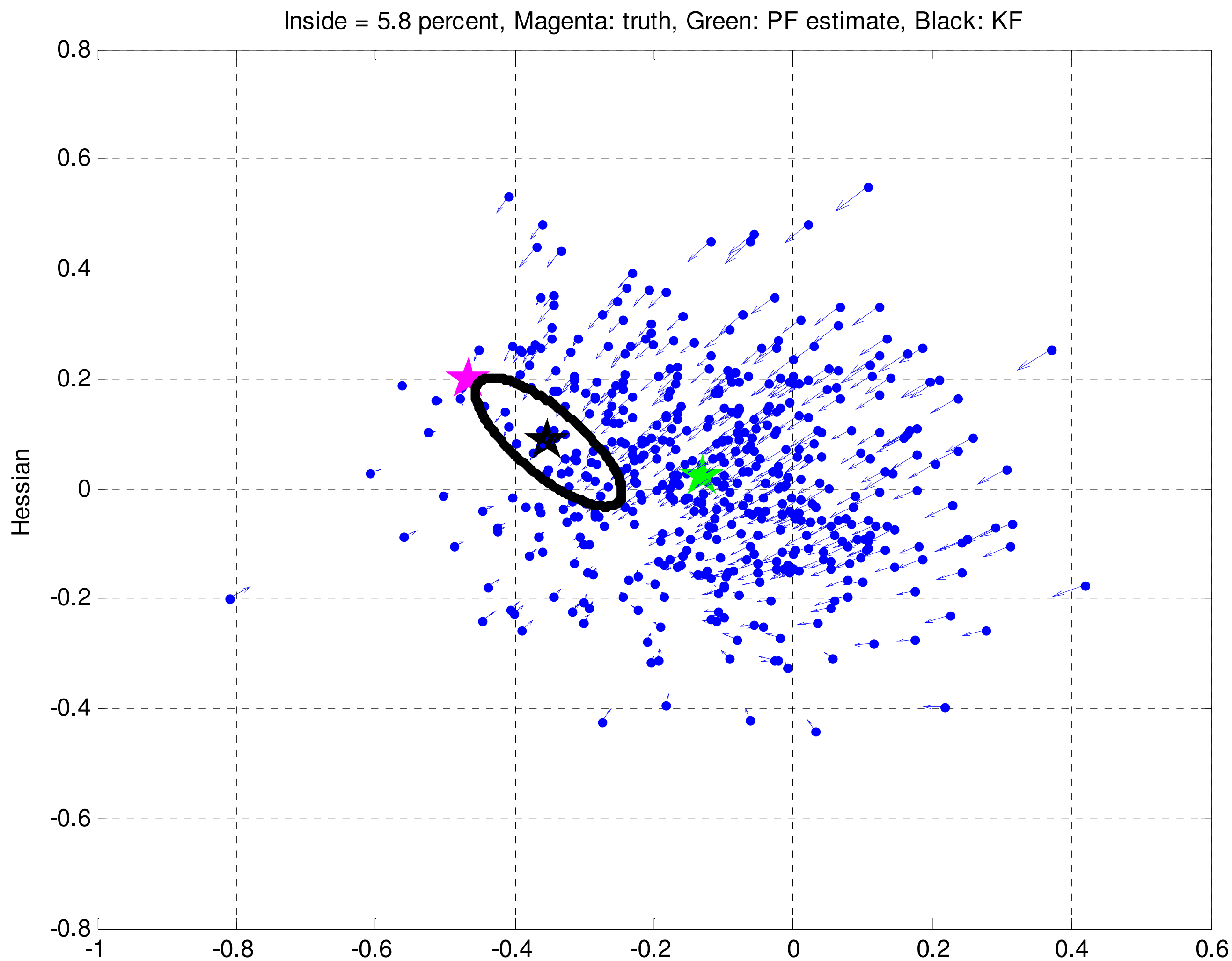

Inside = 27.6 percent, Magenta: truth, Green: PF estimate, Black: KF
Ax+b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

incompressible particle flow

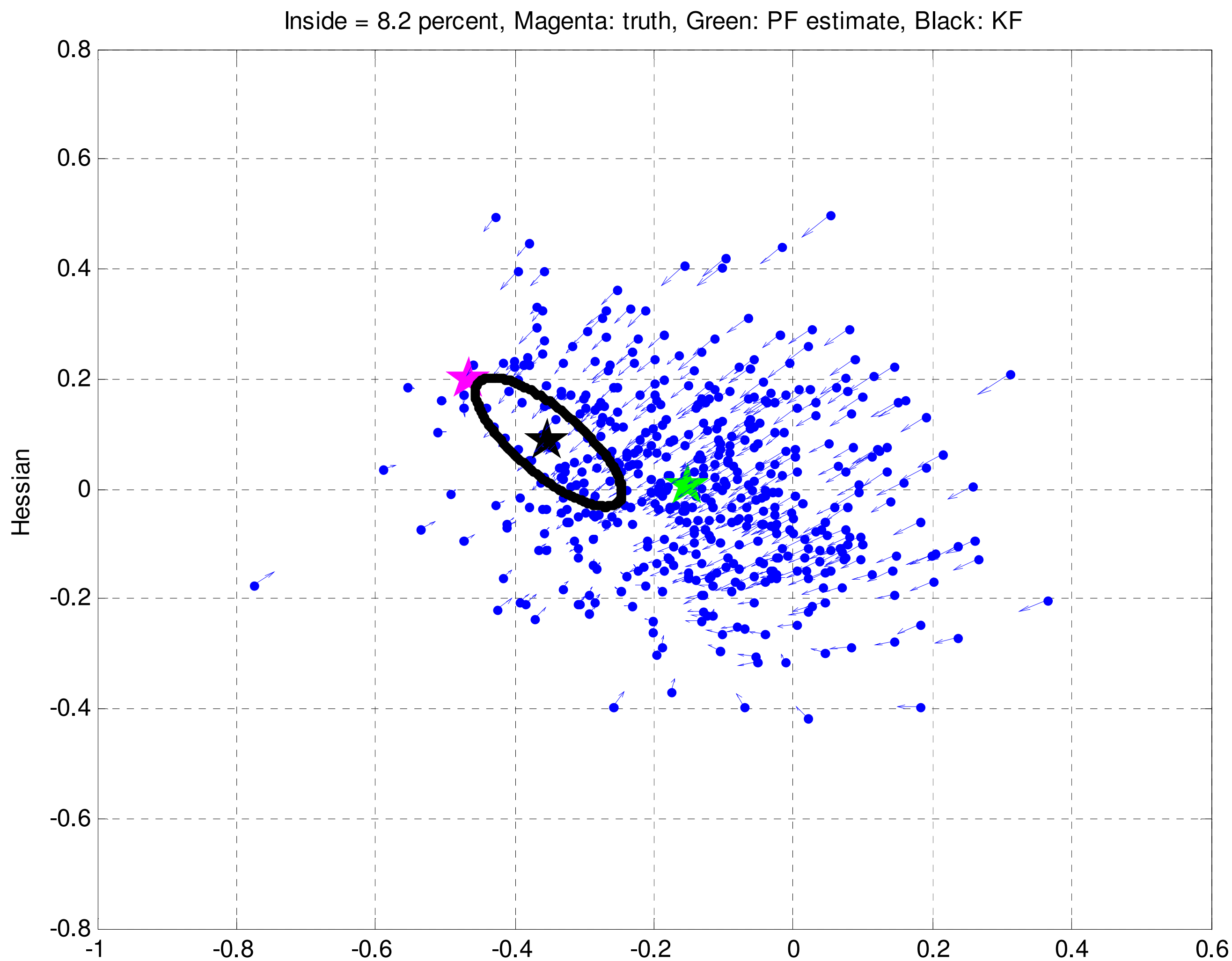
$$\frac{\mathrm{dx}}{\mathrm{d}\lambda} = {-\log(h)\left(\frac{\partial \log p}{\partial x}\right)^T} \Big/ {\left\|\frac{\partial \log p}{\partial x}\right\|^2}$$

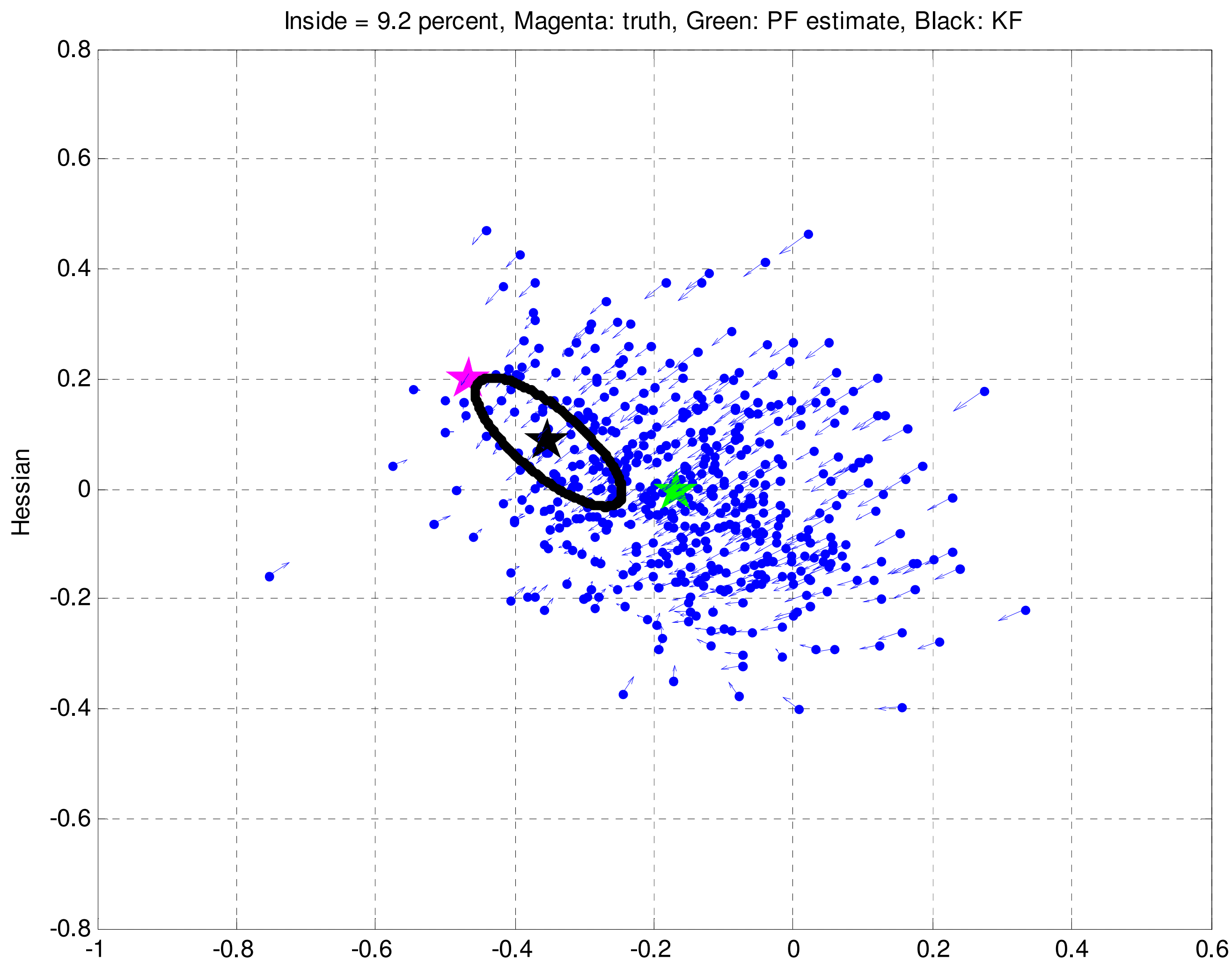
$$\frac{dx}{d\lambda} = 0 \qquad \text{for zero gradient}$$

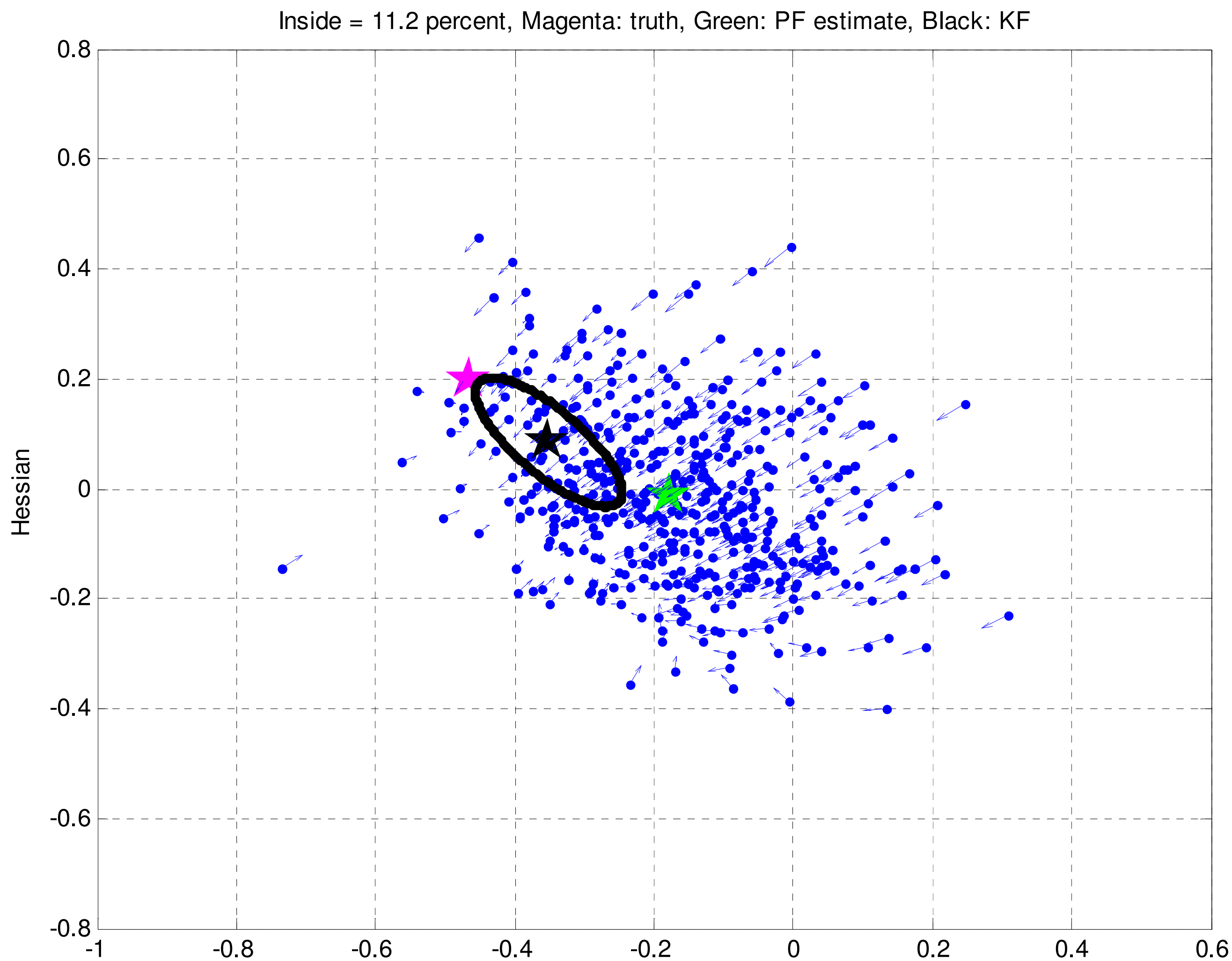

Inside = 6 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

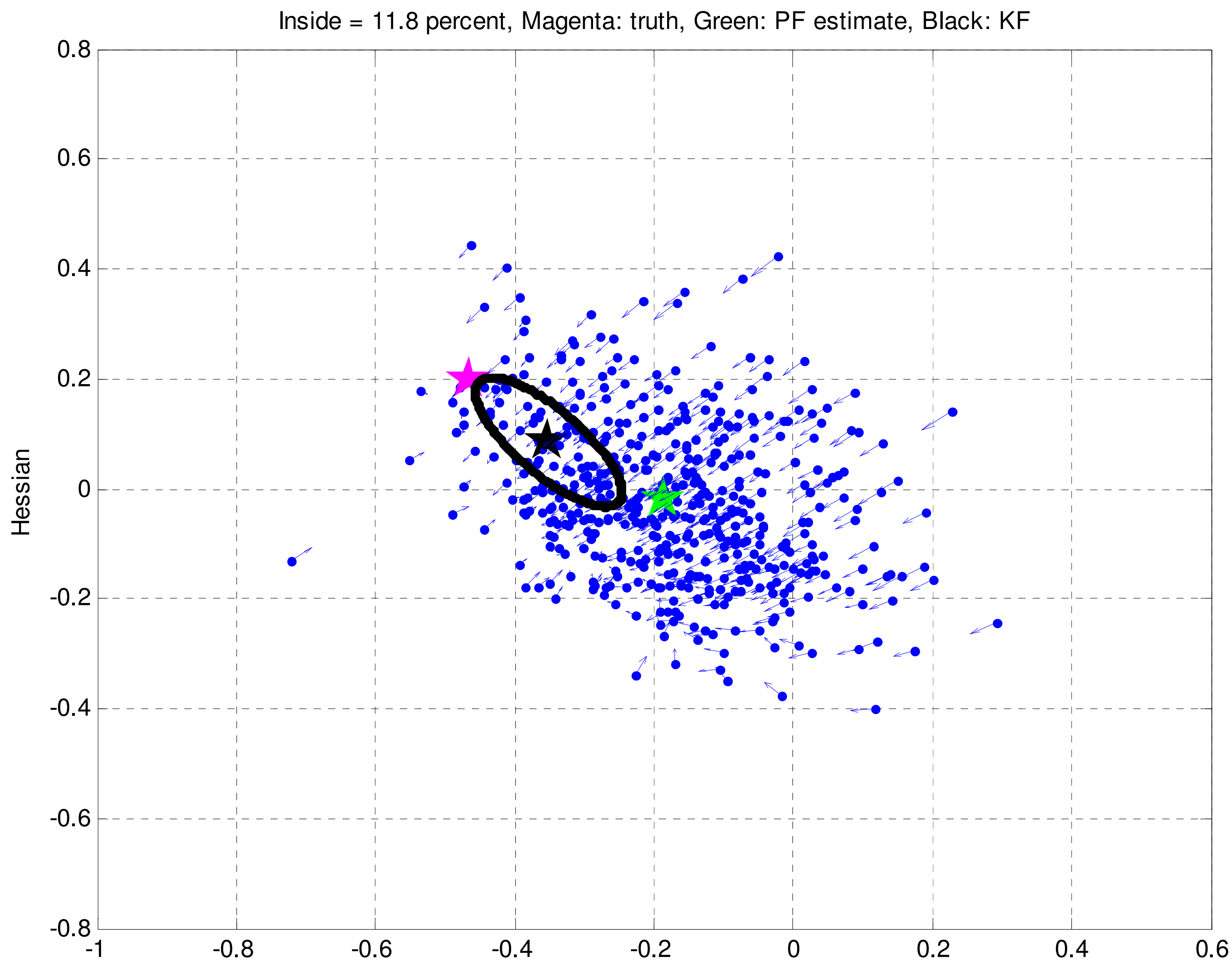


Inside = 5.8 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

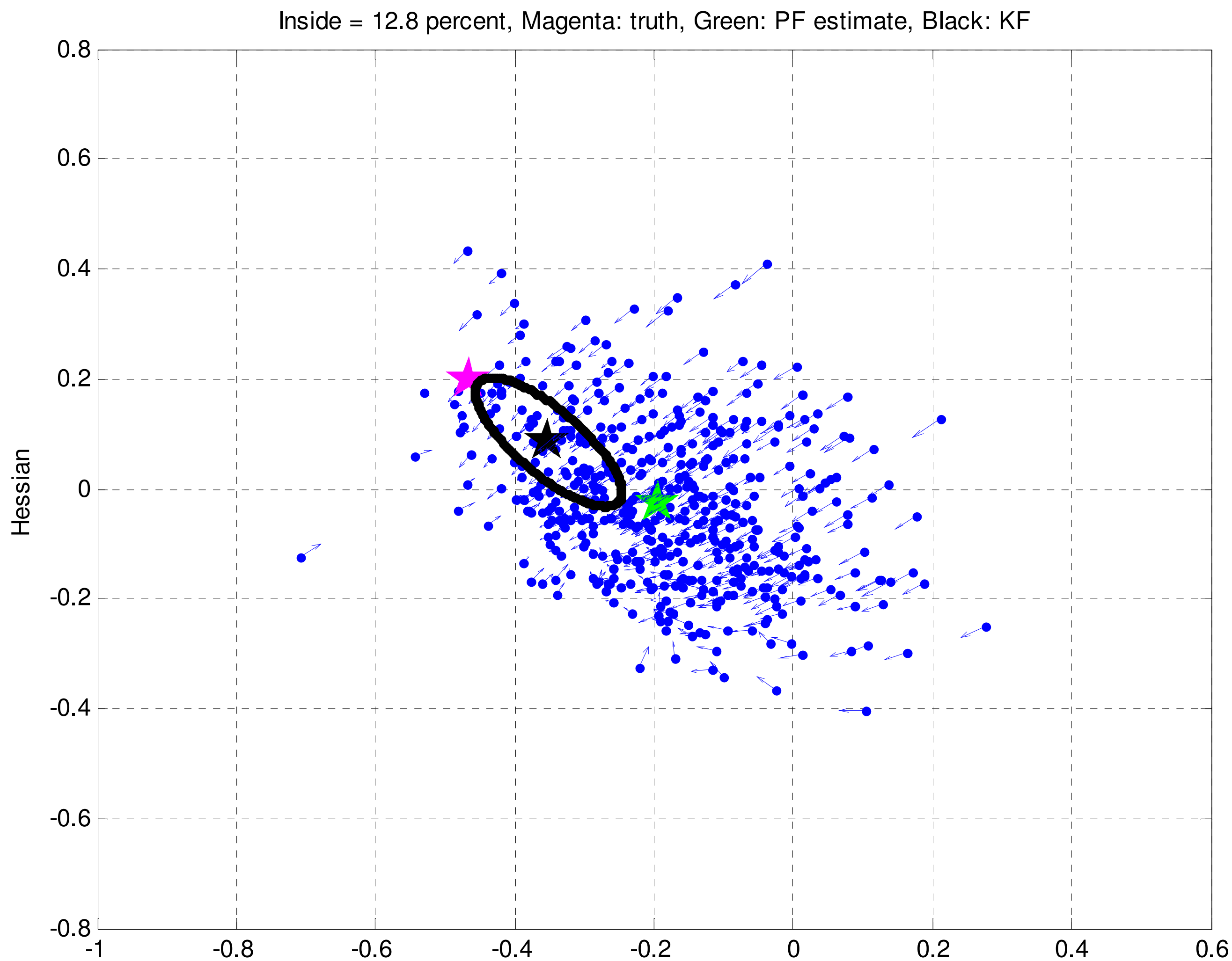

Inside = 8.2 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6


Inside = 9.2 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

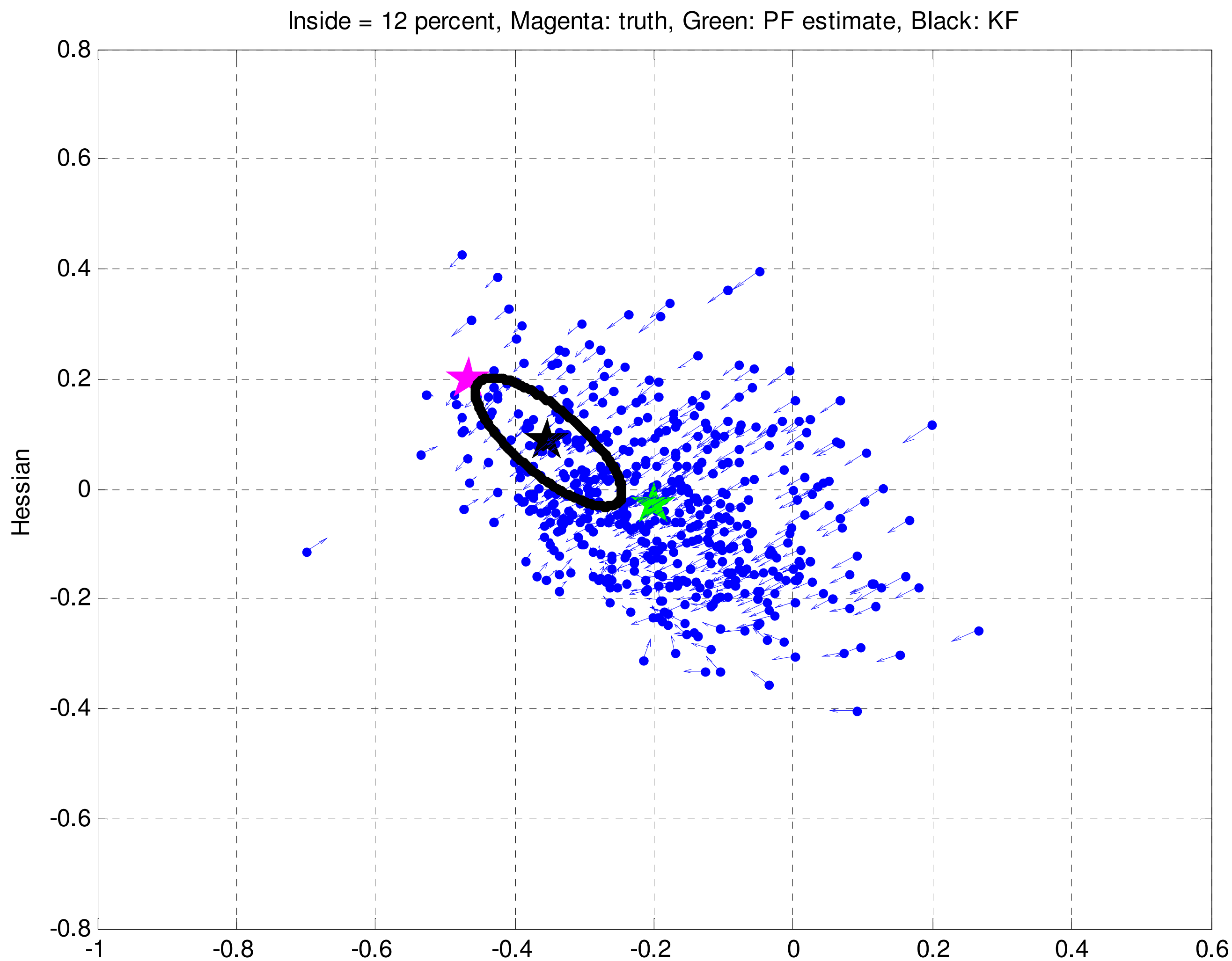

Inside = 11.2 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

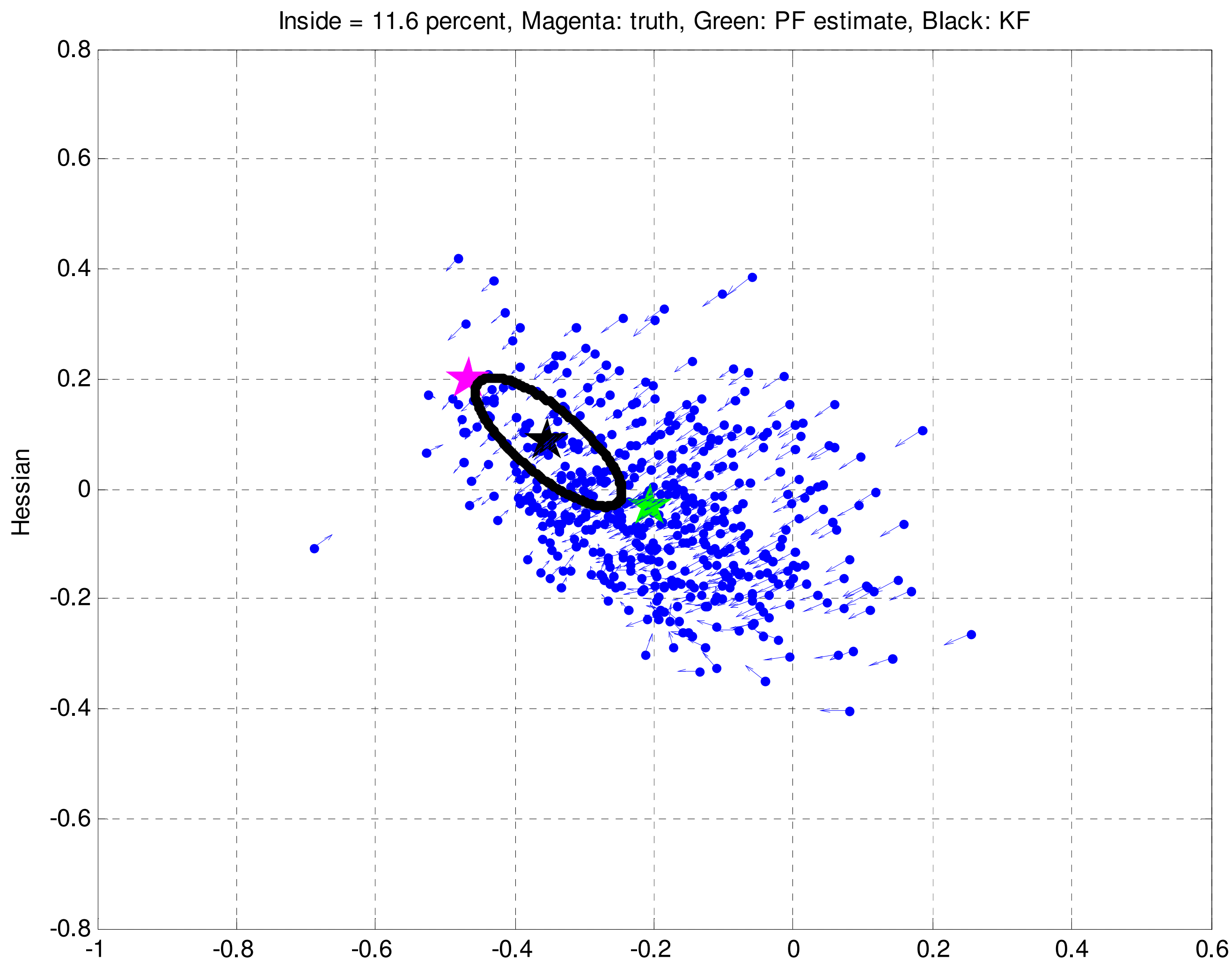


Inside = 11.8 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6



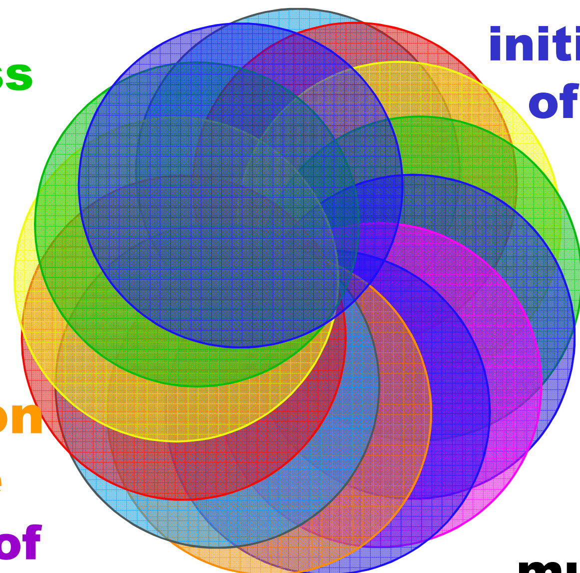
Inside = 12.8 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6


Inside = 12 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

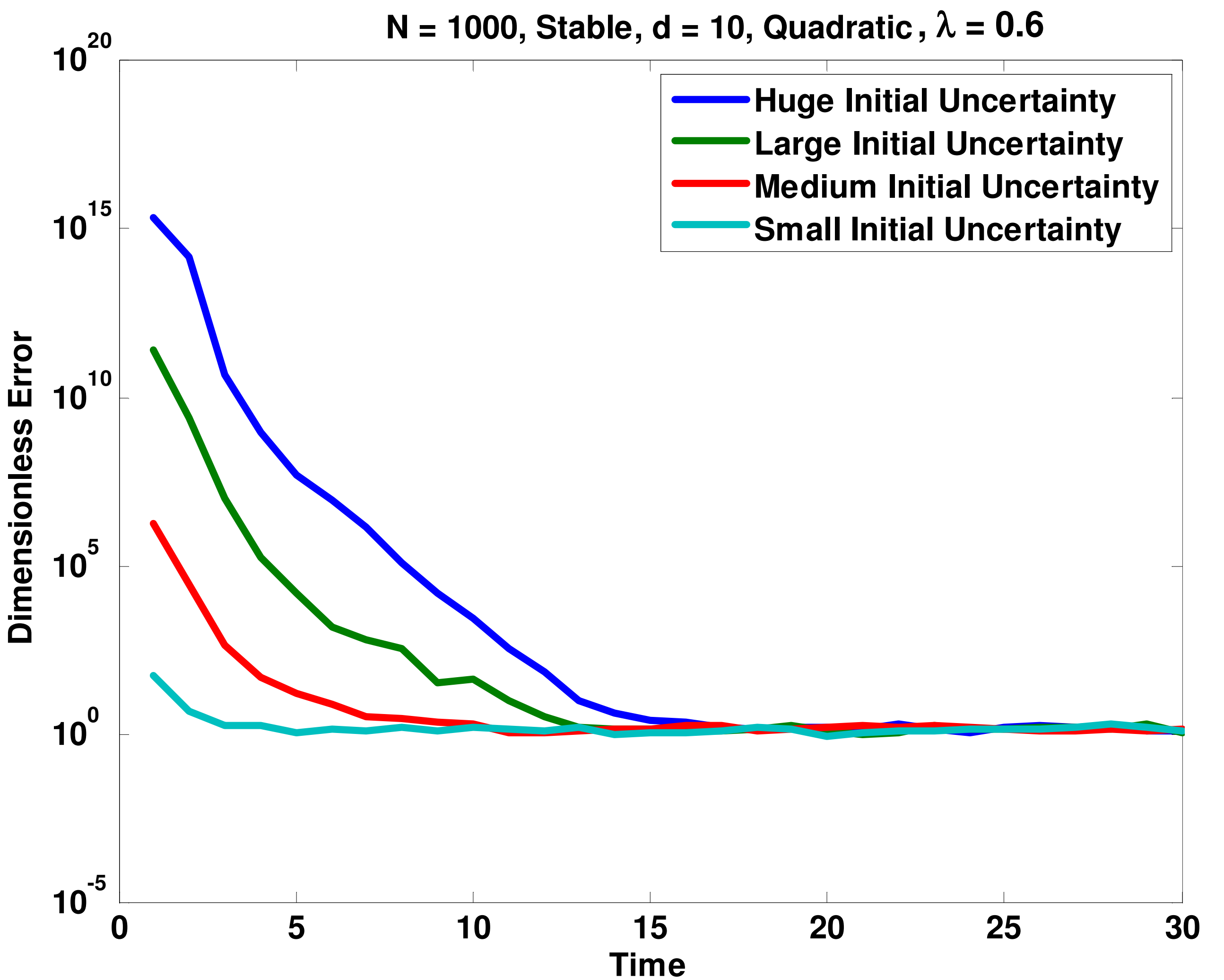


Inside = 11.6 percent, Magenta: truth, Green: PF estimate, Black: KF
Hessian
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

nonlinear filter performance (accuracy wrt optimal & computational complexity)

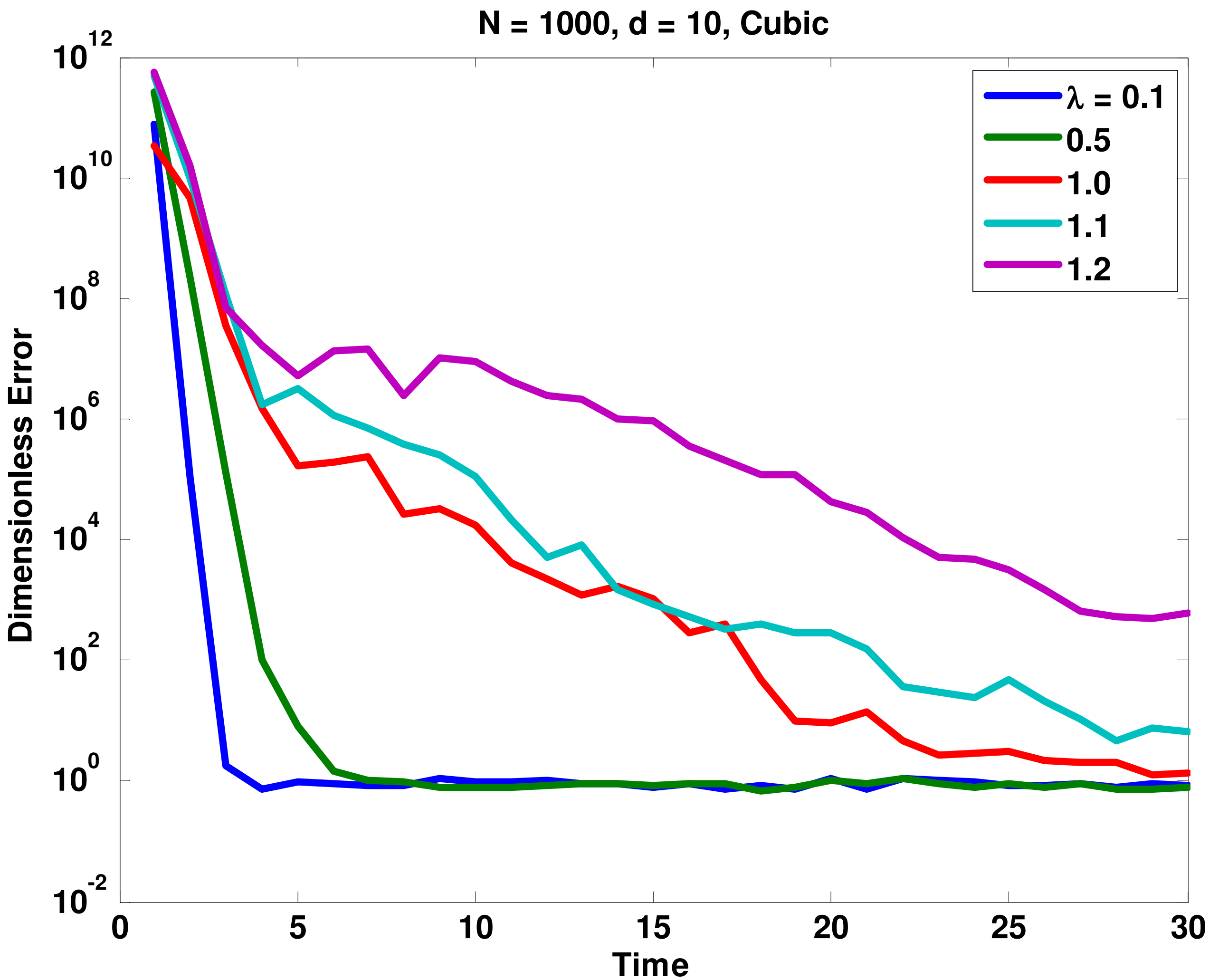


variation in initial uncertainty of x



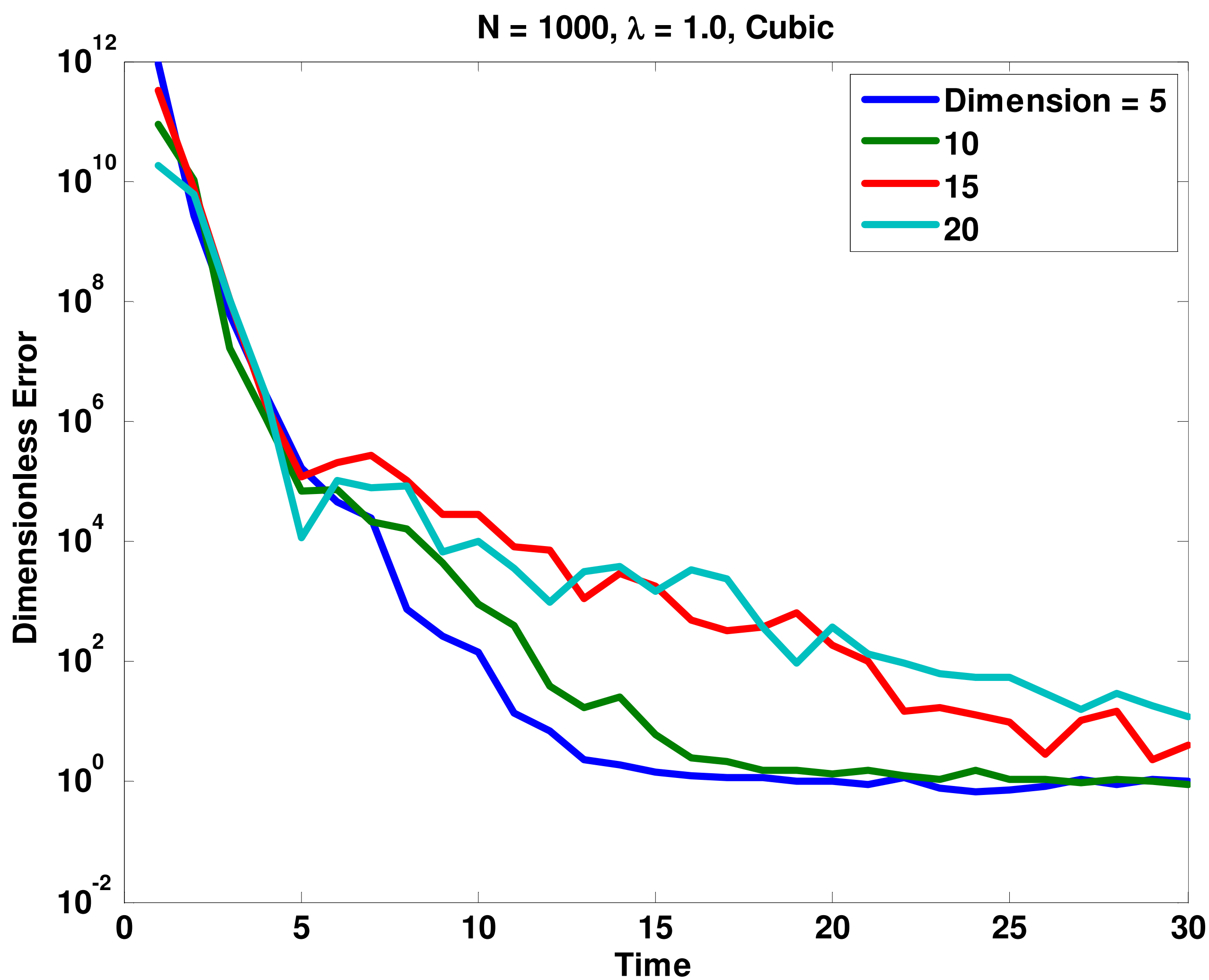
25 Monte Carlo Trials

variation in eigenvalues of the plant (λ)



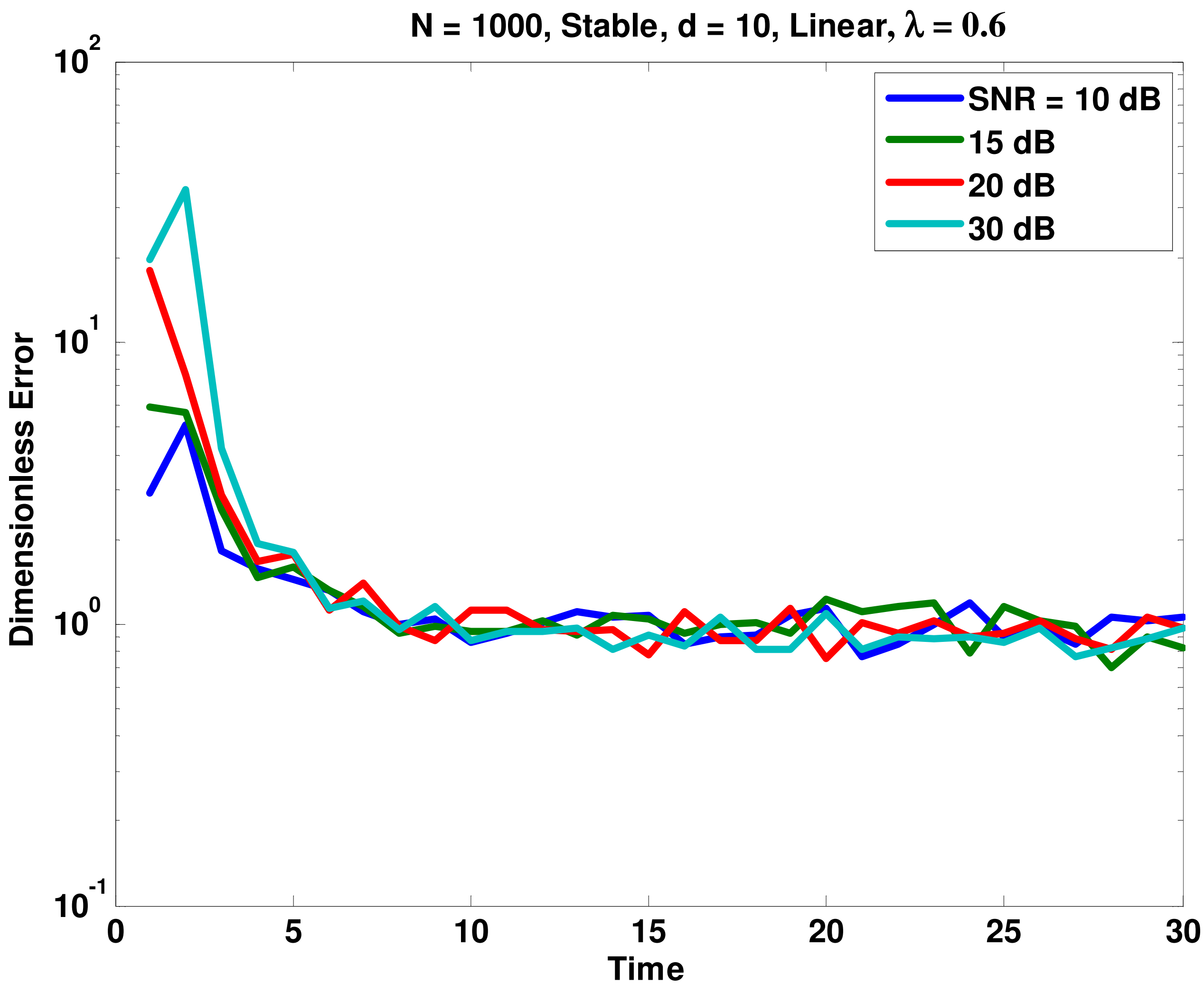
25 Monte Carlo Trials

variation in dimension of x



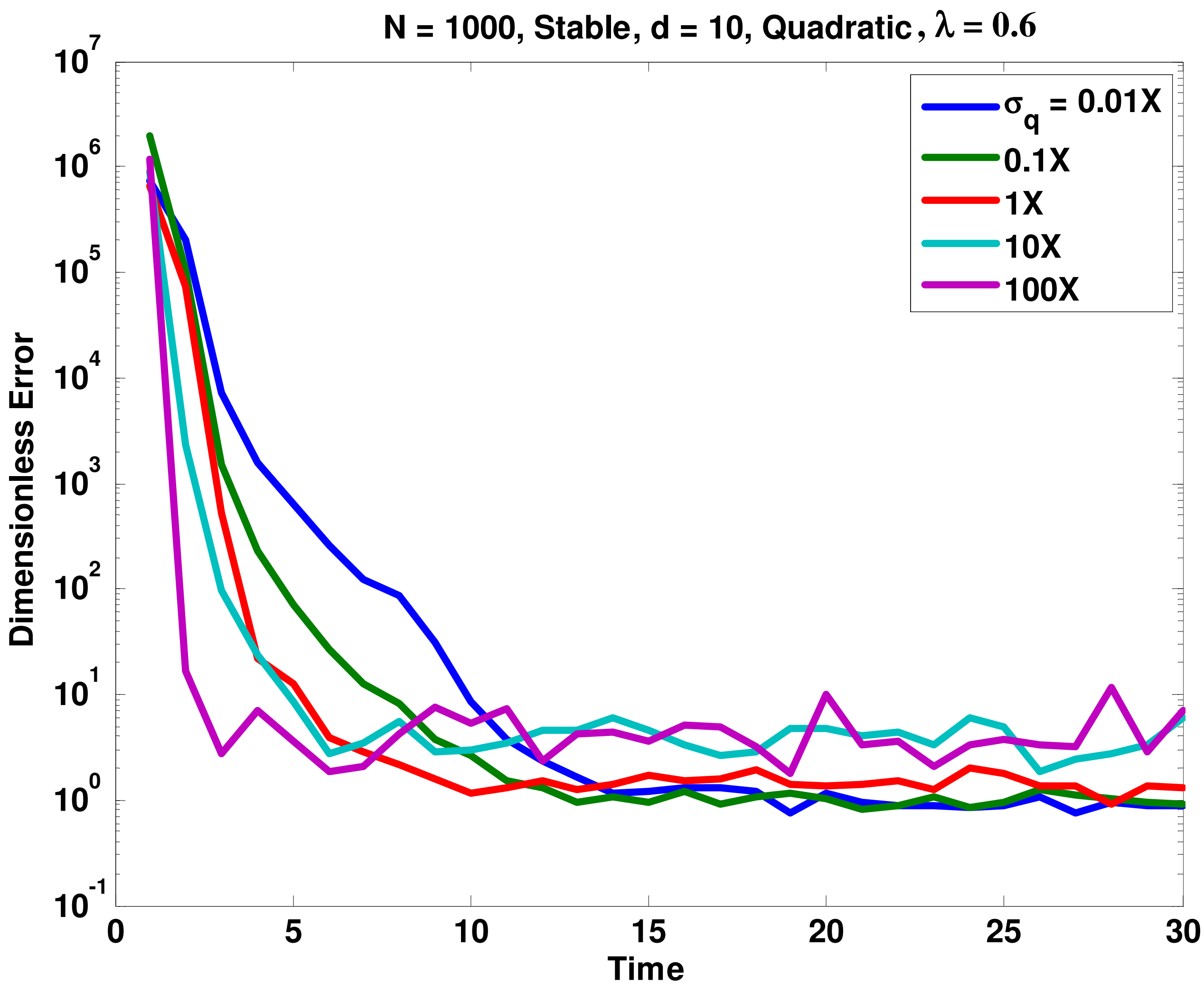
25 Monte Carlo Trials

variation in SNR



25 Monte Carlo Trials

variation in process noise

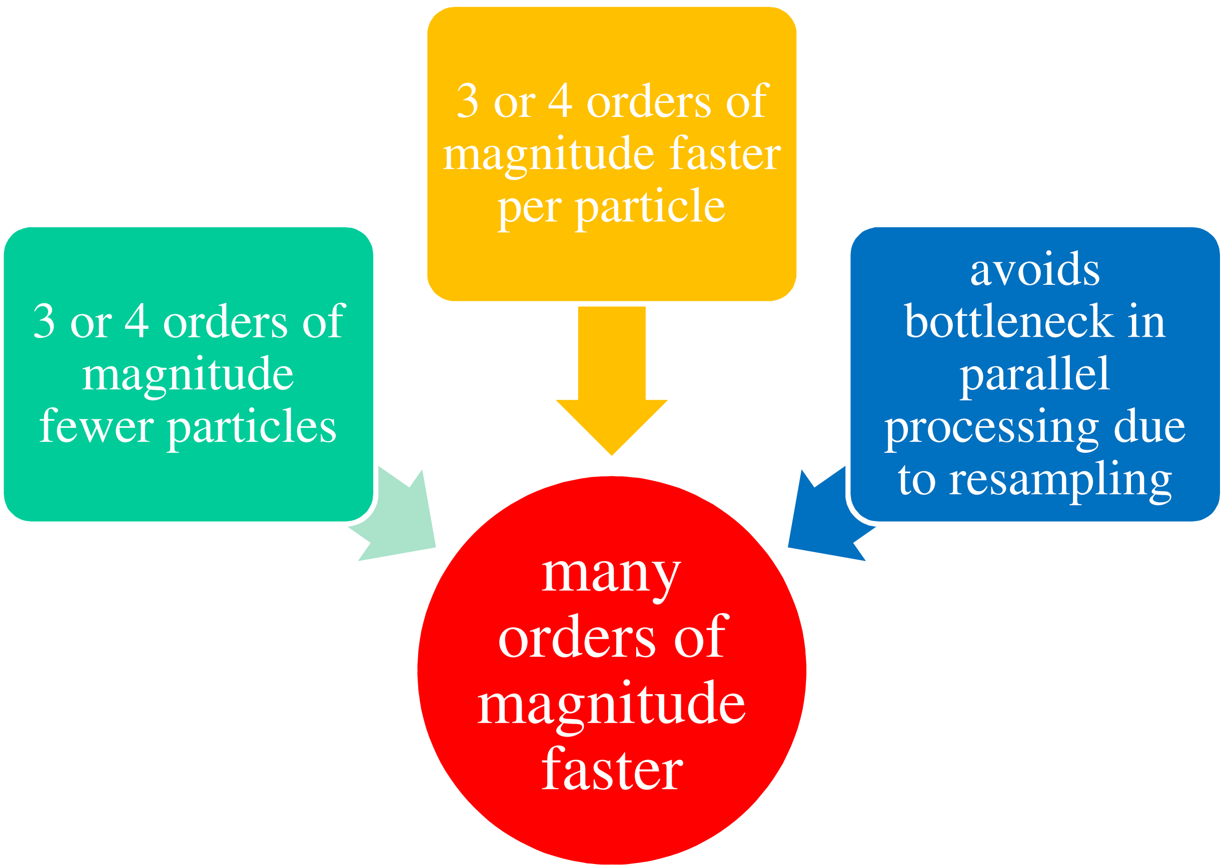


25 Monte Carlo Trials

particle flow filter

- orders of magnitude faster than standard particle filters
- orders of magnitude more accurate than the extended Kalman filter for difficult nonlinear problems
- solves particle degeneracy problem using particle flow induced by log-homotopy for Bayes' rule
- no resampling of particles
- no proposal density
- no importance sampling & no MCMC methods
- unnormalized log probability density
- embarrassingly parallelizable w/o resampling bottleneck (unlike other particle filters)
- exploits smoothness & regularity of densities

particle flow filter is many orders of magnitude faster real time computation (for the same or better estimation accuracy)



History of Mathematics

1. Creation of the integers

2. Invention of counting

3. Invention of addition as a fast method of counting

4. Invention of multiplication as a fast method of addition

5. Invention of particle flow as a fast method of multiplication*

fundamental PDE for exact particle flow:

$$\frac{dx}{d\lambda} = f(x,\lambda)$$

$$\frac{\partial p(x,\lambda)}{\partial \lambda} = -Tr\left[\frac{\partial (pf)}{\partial x}\right]$$

Fokker-Planck equation with Q = 0

$$\frac{\partial \log p(x,\lambda)}{\partial \lambda} p(x,\lambda) = -Tr\left[\frac{\partial (pf)}{\partial x}\right]$$

assume log-homotopy

$$\log p(x,\lambda) = \log g(x) + \lambda \log h(x)$$

$$\log h(x) p(x,\lambda) = -p(x,\lambda) Tr\left[\frac{\partial f}{\partial x}\right] - \frac{\partial p}{\partial x} f$$

$$\log(h) = -div(f) - \frac{\partial \log p}{\partial x} f$$

first order linear underdetermined PDE in $f(x,\lambda)$

direct integration of fundamental PDE:

use the divergence form of the PDE :

$$div(q(x,\lambda)) = Tr\left[\frac{\partial q(x,\lambda)}{\partial x}\right] = \eta(x,\lambda)$$

$$\eta = \frac{\partial q_1}{\partial x_1} + \frac{\partial q_2}{\partial x_2} + \ldots + \frac{\partial q_d}{\partial x_d}$$

$$\frac{\partial q_j}{\partial x_j} = \eta - \sum_{k \neq j}^{d} \frac{\partial q_k}{\partial x_k}$$

$$q_j(x) = \int^{x_j} \left[\eta(x) - \theta_j(x)\right] dx_j$$

pick for best stability of particle flow

$$\theta_j(x) = \sum_{k \neq j}^{d} \frac{\partial q_k}{\partial x_k} = \text{arbitrary function (except for compatibility conditions)}$$

Assuming regularity conditions on η and Ω, a solution for $q(x, \lambda)$ exists

iff $\int_{\Omega} \eta(x)dx = 0.$

more details of direct integration:

$$div(q) = \eta(x)$$

$$q_k = \int^{x_k} \left[\eta(x) - \rho(x_k) \int_{\Omega_k} \eta(x) dx_k \right] dx_k \text{ for k} \geq 2$$

in which

$\rho(\mathrm{x_k}) =$ arbitrary function such that

$$\int_{\Omega_k} \rho(\mathrm{x_k}) dx_k = 1$$

assuming smooth functions with compact support,

and Ω is bounded, open, connected smooth set,

a necessary & sufficient condition for the existence

of a solution q(x) is that $\int_{\Omega} \eta(\mathrm{x}) \mathrm{dx} = 0$

Oh's Formula for Monte Carlo errors

$$\sigma^2 \approx \left\{ \left[\frac{1+k}{\sqrt{1+2k}} \right] \exp\left[\frac{\varepsilon^2}{1+2k} \right] \right\}^d / N$$

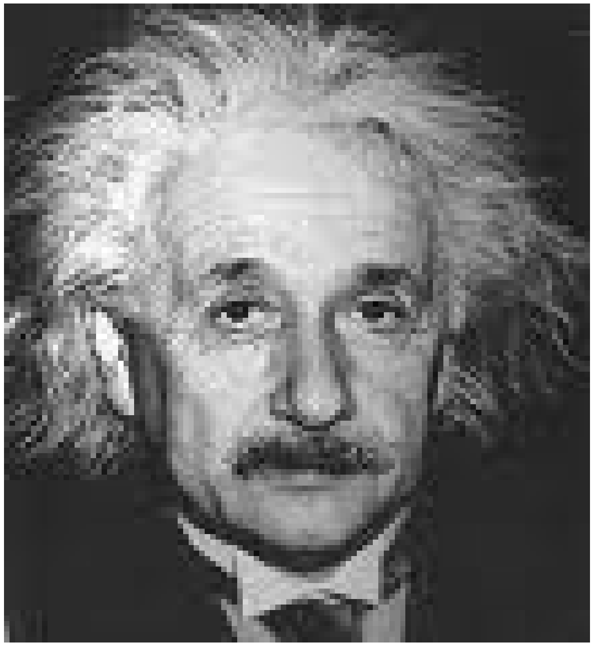
Assumptions:

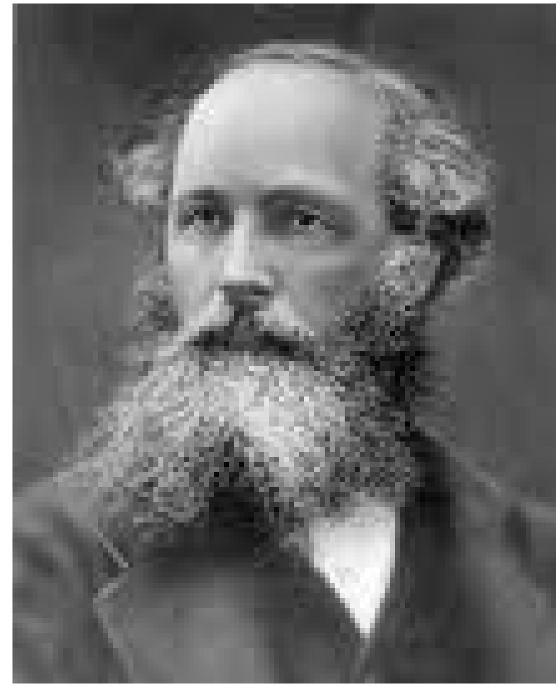
(1) Gaussian density (zero mean & unit covariance matrix)

(2) d-dimensional random variable

(3) Proposal density is also Gaussian with mean ε and covariance matrix kI, but it is not exact for k ≠ 1 or ε ≠ 0

(4) N = number of Monte Carlo trials





1. derive PDE
2. solve PDE
3. test solution

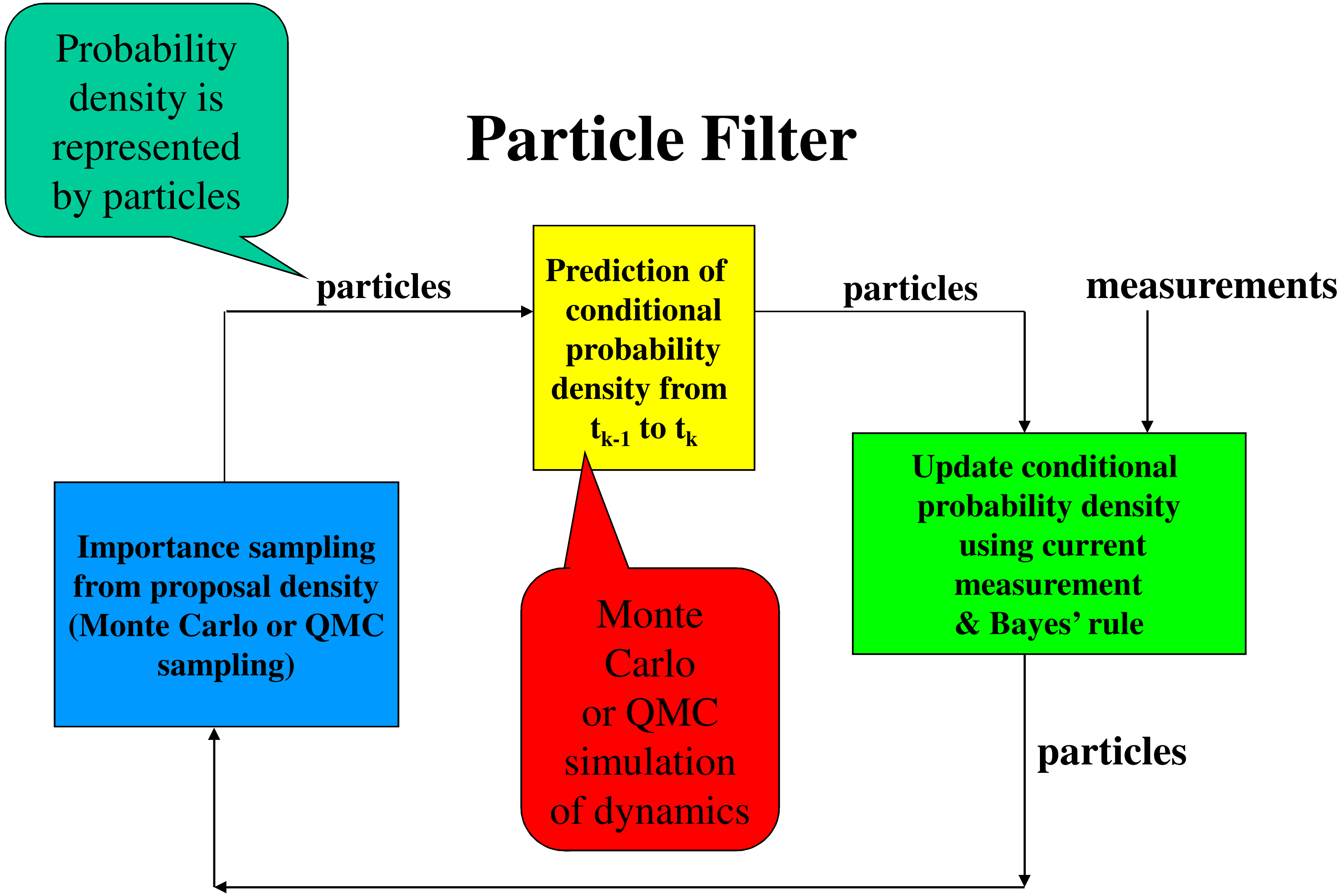
difficulties for exact finite dimensional filters vs. particle filters

	Bayes' update of conditional density of x	prediction of conditional density of x with time
1. exact filters (e.g., Daum 1986)	easy	hard
2. particle filters	hard	easy
3. hybrid of exact & particle filters	?	?

type of nonlinear filter	statistics computed	computational complexity	estimation accuracy	representation of probability density
extended Kalman filters	mean vector & covariance matrix	d^3	sometimes good but often highly suboptimal	mean vector & covariance matrix
unscented Kalman filters	mean vector & covariance matrix	d^3	sometimes better than EKF but sometimes worse	mean vector & covariance matrix
batch least squares	mean vector & covariance matrix	d^3	sometimes better than EKF but sometimes worse	mean vector & covariance matrix
numerical solution of Fokker-Planck PDE	full conditional probability density of state	curse of dimensionality	optimal*	points in state space and/or smooth functions
particle filters	full conditional probability density of state	curse of dimensionality	optimal*	particles
exact recursive filters (Kalman, Beneš, Daum, Wonham, Yau)	full conditional probability density of state	polynomial in d (for special problems)	optimal (for special problems)	sufficient statistics

What is a particle filter?

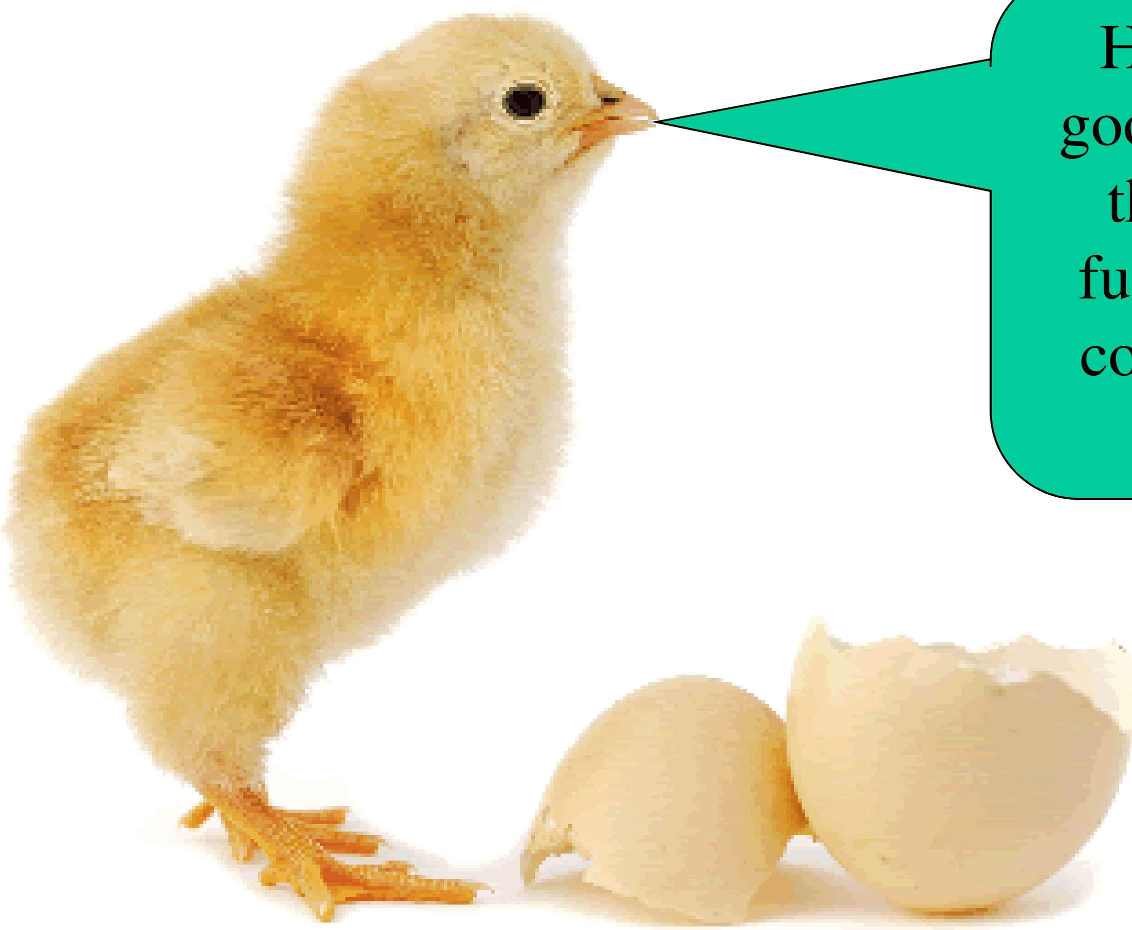
Particle Filter



Why engineers like particle filters:

- Very easy to code
- Extremely general dynamics & measurements: nonlinear & non-Gaussian
- Optimal estimation accuracy (if you use enough particles….)
- You don't need to know anything about stochastic differential equations or any fancy numerical methods for solving PDEs
- Some people (erroneously) think that PFs beat the curse of dimensionality

chicken & egg problem



How do you pick a good way to represent the product of two functions before you compute the product itself?