

# Inference in non-linear and non-Gaussian models

Mattias Villani

**Department of Statistics  
Stockholm University**



# Taxonomy of state-space models

## ■ Linear Gaussian models

Measurement model:  $\mathbf{y}_t = \mathbf{C}\mathbf{z}_t + \varepsilon_t, \quad \varepsilon_t \stackrel{\text{iid}}{\sim} N(\mathbf{0}, \Sigma_\varepsilon)$

Transition model:  $\mathbf{z}_t = \mathbf{A}\mathbf{z}_{t-1} + \eta_t, \quad \eta_t \stackrel{\text{iid}}{\sim} N(\mathbf{0}, \Sigma_\eta)$

## ■ Non-linear (additive) Gaussian models

Measurement model:  $\mathbf{y}_t = \mathbf{C}(\mathbf{z}_t) + \varepsilon_t, \quad \varepsilon_t \stackrel{\text{iid}}{\sim} N(\mathbf{0}, \Sigma_\varepsilon)$

Transition model:  $\mathbf{z}_t = \mathbf{A}(\mathbf{z}_{t-1}) + \eta_t, \quad \eta_t \stackrel{\text{iid}}{\sim} N(\mathbf{0}, \Sigma_\eta)$

## ■ General distribution (non-linear and non-Gaussian) models

Measurement model:  $\mathbf{y}_t \sim p(\mathbf{y}_t | \mathbf{z}_t)$

Transition model:  $\mathbf{z}_t \sim p(\mathbf{z}_t | \mathbf{z}_{t-1})$

## ■ Hybrids: nonlinear measurement + linear Gaussian transition.

# Poisson time series for counts

## ■ Poisson model with time-varying intensity

$$y_t | z_t \sim \text{Poisson}(\exp(z_t))$$

$$z_t = \mu + \phi(z_{t-1} - \mu) + \eta_t, \quad \eta_t \stackrel{\text{iid}}{\sim} N(0, \sigma_\eta^2)$$



# Common Bayesian computational approaches

- **Kalman** filtering and **FFBS sampling** for linear Gaussian models. [1, 2]
- **Particle MCMC** and **SMC** [3]
- **Hamiltonian Monte Carlo** (Stan/Turing.jl) [4]
- **Variational approximations** [5]
- **INLA** [6]
- **Gaussian approximations** [7]

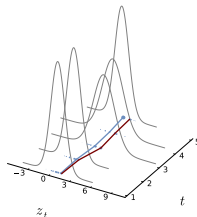
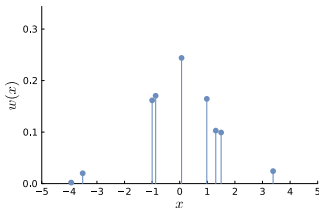
# Particle filters

- Approximate filtering posteriors by **weighted particle system**

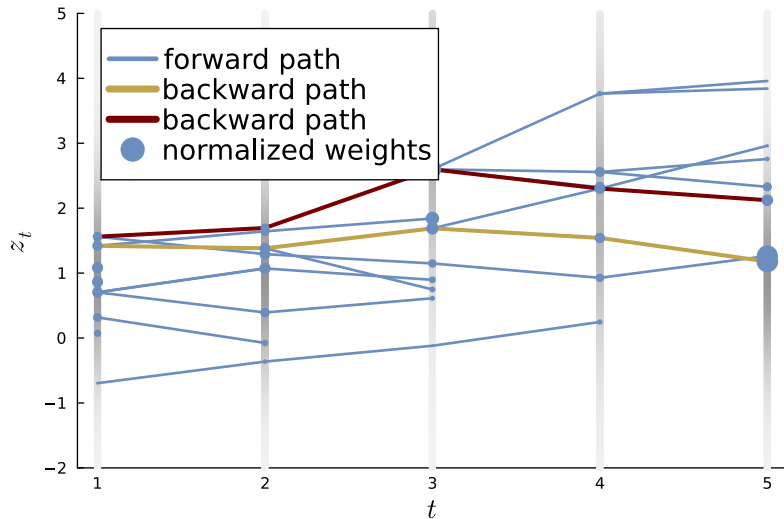
$$\hat{p}(\mathbf{z}_t | \mathbf{y}_{1:t}) = \sum_{j=1}^M \omega_t^{(j)} \delta_{\mathbf{z}_t^{(j)}}(\mathbf{z}_t)$$

- **Sequential importance sampling** moves particles over time.
- **Bootstrap filter**: importance density is the prior at time  $t$ :

$$q(\mathbf{z}_t | \mathbf{y}_{1:t}) = p(\mathbf{z}_t | \mathbf{y}_{1:t-1})$$



# Particle filters



# The Bootstrap filter

## Bootstrap filter update for general state-space model

$$p(\mathbf{z}_{t-1}|\mathbf{y}_{1:t-1}) \Rightarrow p(\mathbf{z}_t|\mathbf{y}_{1:t-1}) \Rightarrow p(\mathbf{z}_t|\mathbf{y}_{1:t})$$

**Input:** particles from previous step  $\mathbf{z}_{t-1}^{(1)}, \dots, \mathbf{z}_{t-1}^{(M)}$   
measurement  $\mathbf{y}_t$   
control signal  $\mathbf{u}_t$

**for**  $j$  in  $1:M$  **do**

    Prior propagation

    draw state particle  $\mathbf{z}_t^{(j)} \sim g(\mathbf{z}_t|\mathbf{z}_{t-1}^{(j)})$

    Measurement update

    compute unnormalized weight  $\tilde{w}_t^{(j)} = p(\mathbf{y}_t|\mathbf{z}_t^{(j)}, \boldsymbol{\theta})$

**end**

normalize weights

$$w_t^{(j)} = \tilde{w}_t^{(j)} / \sum_{k=1}^M \tilde{w}_t^{(k)} \text{ for } j = 1, \dots, M$$

Resampling

**for**  $j$  in  $1:M$  **do**

$k_j \sim \text{Cat}(w_t^{(1)}, \dots, w_t^{(M)})$   
     $\mathbf{z}_t^{(j)} \leftarrow \mathbf{z}_t^{(k_j)}$

**end**

**Output:**  $M$  draws  $\mathbf{z}_t^{(1)}, \dots, \mathbf{z}_t^{(M)}$  from  $p(\mathbf{z}_t|\mathbf{y}_{1:t})$ .

# Particle Gibbs with Ancestor Sampling (PGAS)

- Particle Gibbs samples from the joint smoothing posterior  $p(\mathbf{z}_{1:T} | \mathbf{y}_{1:T}, \boldsymbol{\theta})$  via a particle filter inside a Gibbs sampler. [8]
- **Conditional particle filter** that conditions on a **reference particle trajectory** from the previous Gibbs iteration.
- Markov kernel that leaves the target posterior invariant.
- Extended target with state + SMC randomness.
- Partially collapsed (SMC randomness integrated out).
- **Particle Gibbs with Ancestor Sampling (PGAS)** [9]
  - ▶ Resamples the reference particle ancestor at every time step.
  - ▶ Mitigates degeneracy and gives better mixing.
  - ▶ Smaller number of particles needed, but more costly.

# Extended Kalman filter (EKF)

- State-space model (linear transition, nonlinear measurement)

$$\mathbf{z}_t = \mathbf{A}\mathbf{z}_{t-1} + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim N(\mathbf{0}, \Sigma_{\eta})$$

$$y_t = \mathbf{c}(\mathbf{z}_t) + \varepsilon_t, \quad \varepsilon_t \sim N(0, \Sigma_{\varepsilon})$$

- Linearized model

$$\mathbf{z}_t = \mathbf{A}\mathbf{z}_{t-1} + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim N(\mathbf{0}, \Sigma_{\eta})$$

$$y_t = \mathbf{c}'(\mathbf{z}_t) + \varepsilon_t, \quad \varepsilon_t \sim N(0, \Sigma_{\varepsilon})$$

## Standard Kalman filter

# Prior propagation step

$$\bar{\boldsymbol{\mu}} = \mathbf{A} * \boldsymbol{\mu} . + \mathbf{B} * \mathbf{u}$$

$$\bar{\boldsymbol{\Omega}} = \mathbf{A} * \boldsymbol{\Omega} * \mathbf{A}' + \Sigma_n$$

# Measurement update

$$\mathbf{K} = \bar{\boldsymbol{\Omega}} * \mathbf{C}' / (\mathbf{C} * \bar{\boldsymbol{\Omega}} * \mathbf{C}' . + \Sigma_e)$$

$$\boldsymbol{\mu} = \bar{\boldsymbol{\mu}} + \mathbf{K} * (\mathbf{y} . - \mathbf{C} * \bar{\boldsymbol{\mu}})$$

$$\boldsymbol{\Omega} = (\mathbf{I} - \mathbf{K} * \mathbf{C}) * \bar{\boldsymbol{\Omega}}$$

## Extended Kalman filter

# Prior propagation step

$$\bar{\boldsymbol{\mu}} = \mathbf{A} * \boldsymbol{\mu} + \mathbf{B} * \mathbf{u}$$

$$\bar{\boldsymbol{\Omega}} = \mathbf{A} * \boldsymbol{\Omega} * \mathbf{A}' + \Sigma_n$$

$$\bar{\mathbf{C}} = \partial \mathbf{C}(\bar{\boldsymbol{\mu}}, \text{Cargs})$$

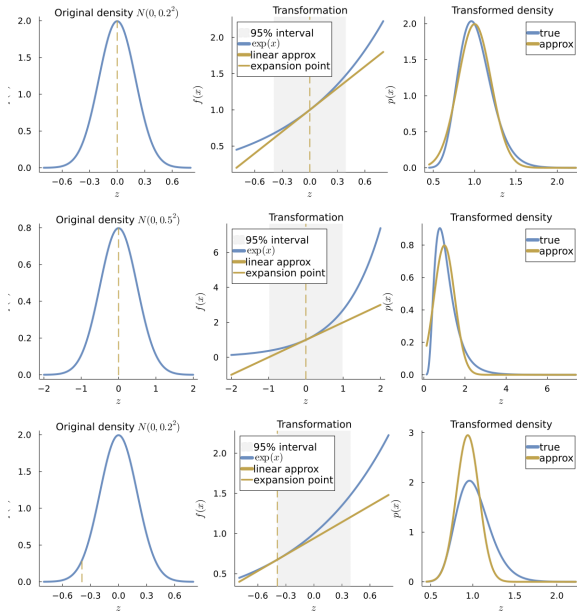
# Measurement update

$$\mathbf{K} = \bar{\boldsymbol{\Omega}} * \bar{\mathbf{C}}' / (\bar{\mathbf{C}} * \bar{\boldsymbol{\Omega}} * \bar{\mathbf{C}}' . + \Sigma_e)$$

$$\boldsymbol{\mu} = \bar{\boldsymbol{\mu}} + \mathbf{K} * (\mathbf{y} . - \mathbf{C}(\bar{\boldsymbol{\mu}}, \text{Cargs}))$$

$$\boldsymbol{\Omega} = (\mathbf{I} - \mathbf{K} * \bar{\mathbf{C}}) * \bar{\boldsymbol{\Omega}}$$

# Extended Kalman filter



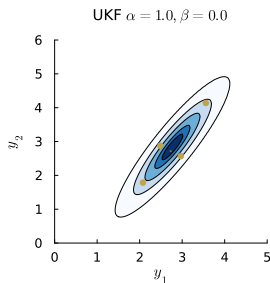
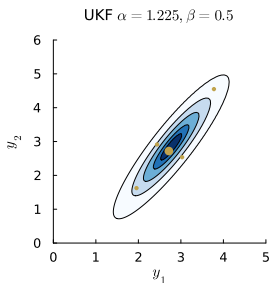
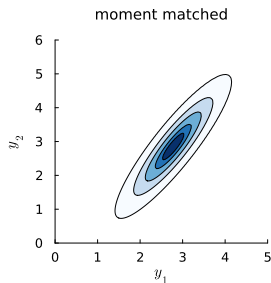
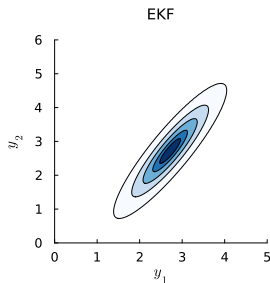
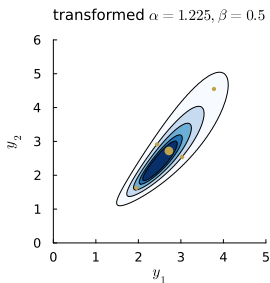
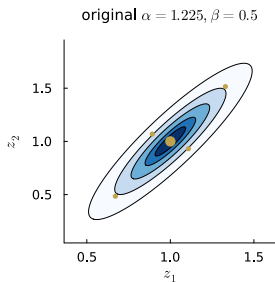
# FFBS-EKF

- EKF: **automatic differentiation** makes it all easy.
- Forward filtering with EKF gives  $\mu_{t|t}, \Omega_{t|t}, \mu_{t|t-1}, \Omega_{t|t-1}$ .
- **Sampling from joint smoothing density** by backward sampling using same multivariate normal distributions as in FFBS.
- Much faster than PGAS.
- **Robust to near-degeneracy** in the transition model.  
Important when global-local shrinkage process priors is used for the parameter evolution. See Lecture 4.

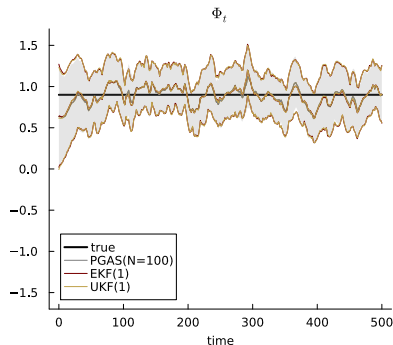
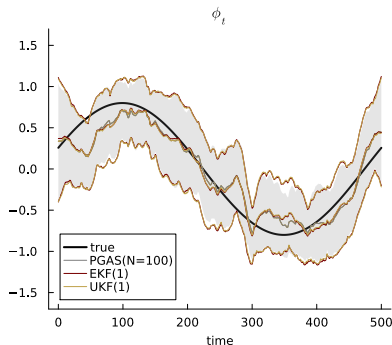
# Alternative approximate Gaussian filtering

- **Moment matching** by numerical integration. [7]
- **Unscented Kalman Filter (UKF)**: map sigma points through nonlinear transition and measurement functions. [10]
- **Iterated EKF** and **Iterated UKF**: expands around posterior mean instead of prior mean. [11, 12]
- Iterated EKF/UKF with **line search**. [11, 12]
- **Prior Linearization filter**. Linearizes  $\mathbf{y} = \mathbf{c}(\mathbf{x})$  by minimizing  $\mathbb{E}_{\text{prior}} \left( \|\mathbf{c}(\mathbf{x}) - (\mathbf{c}_0 + \mathbf{C}\mathbf{x})\|^2 \right)$ . [13]
- **Posterior linearization filter** minimizes  $\mathbb{E}_{\text{post}} \left( \|\mathbf{c}(\mathbf{x}) - (\mathbf{c}_0 + \mathbf{C}\mathbf{x})\|^2 \right)$  by iterating. [13]
- **Laplace approximation**: approximates by posterior mode and inverse negative Hessian. [14]

# Gaussian approximations to bivariate log-normal



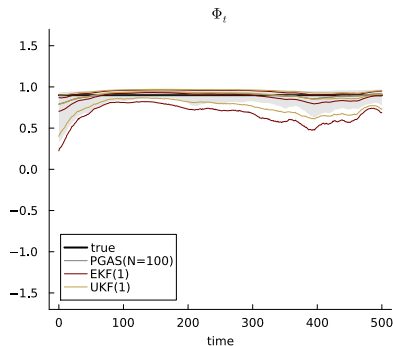
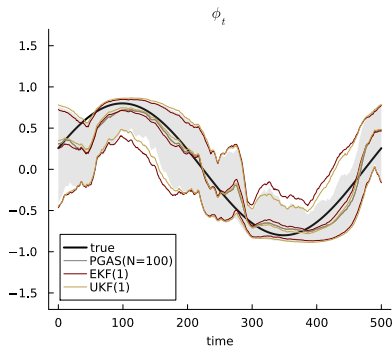
# Seasonal AR - without stability restrictions









## ■ Compute time 10000 draws:

- ▶ PGAS(100): 211.7 sec
- ▶ FFBS-EKF: 11.9 sec
- ▶ FFBS-UKF: 14.7 sec

# Seasonal AR - with stability restrictions



- This can be improved by alternative stability parameterizations.

-  C. K. Carter and R. Kohn, "On gibbs sampling for state space models," *Biometrika*, vol. 81, no. 3, pp. 541–553, 1994.
-  S. Frühwirth-Schnatter, "Data augmentation and dynamic linear models," *Journal of time series analysis*, vol. 15, no. 2, pp. 183–202, 1994.
-  N. Chopin, O. Papaspiliopoulos, *et al.*, *An introduction to sequential Monte Carlo*, vol. 4. Springer, 2020.
-  R. M. Neal, "Mcmc using hamiltonian dynamics," *Handbook of markov chain monte carlo*, vol. 2, no. 11, p. 2, 2011.
-  L. S. Tan and D. J. Nott, "Gaussian variational approximation with sparse precision matrices," *Statistics and Computing*, vol. 28, no. 2, pp. 259–275, 2018.
-  H. Rue, S. Martino, and N. Chopin, "Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations," *Journal of the Royal Statistical*

*Society Series B: Statistical Methodology*, vol. 71, no. 2, pp. 319–392, 2009.



S. Särkkä and L. Svensson, *Bayesian filtering and smoothing*, vol. 17.

Cambridge university press, 2023.



C. Andrieu, A. Doucet, and R. Holenstein, “Particle markov chain monte carlo methods,” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 72, no. 3, pp. 269–342, 2010.



F. Lindsten, M. Jordan, and T. B. Schön, “Particle gibbs with ancestor sampling,” *Journal of Machine Learning Research*, vol. 15, p. 2145–2184, 06 2014.



S. J. Julier and J. K. Uhlmann, “Unscented filtering and nonlinear estimation,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.



M. A. Skoglund, G. Hendeby, and D. Axehill, “Extended kalman filter modifications based on an optimization view

point,” in *2015 18th International Conference on Information Fusion (Fusion)*, pp. 1856–1861, IEEE, 2015.



M. A. Skoglund, F. Gustafsson, and G. Hendeby, “On iterative unscented kalman filter using optimization,” in *2019 22th International Conference on Information Fusion (FUSION)*, pp. 1–8, IEEE, 2019.



Á. F. García-Fernández, L. Svensson, M. R. Morelande, and S. Särkkä, “Posterior linearization filter: Principles and implementation using sigma points,” *IEEE transactions on signal processing*, vol. 63, no. 20, pp. 5561–5573, 2015.



S. Koyama, L. Castellanos Pérez-Bolde, C. R. Shalizi, and R. E. Kass, “Approximate methods for state-space models,” *Journal of the American Statistical Association*, vol. 105, no. 489, pp. 170–180, 2010.