Hidden Markov Models presentation

Electricity Load Forecasting

using Particle Filters

Anas ATMANI, Maxime CALLOIX and Romane PERSCH

January 11th, 2017

Summary

Context

Method

Results

Conclusion

Context

Context / Data

Application of Sequential Monte Carlo methods & more specifically Particle Filters to
Electricity load forecasting for EDF.

Adding a new observation does not require the model to be entirely re-estimated.(vs. traditional MCMC methods such as Gibbs sampling).

Electricity Consumption data every 30min in Metropolitan France (ex. Corsica) over the 2011-2015 period from **RTE**

Average Temperature of weather stations, **every 3h** from **Météo France** (while the article uses EDF Internal Forecast Model)

Model (for a chosen half-hour)

Electricity load at time n

State

Error

$$\boxed{y_n} = \boxed{x_n} + \boxed{\nu_n} \quad \text{where } \nu_n \sim \mathcal{N}(0, \sigma^2)$$

The state x_n is made of 2 parts :

$$x_n = \boxed{x_n^{season}} + \boxed{x_n^{heat}}$$

We take into account the different daytypes (week-ends, holidays, days before/after holidays,...)

We take into account the heating, but not the cooling

which are defined by :

$$x_n^{season} = s_n \cdot \kappa_{daytype_n}$$

$$x_n^{heat} = g_n^{heat}(T_n^{heat} - u^{heat})\boxed{\mathbb{1}_{(u^{heat} > T_n^{heat})}}$$

g^{heat} is used only when the temperature is under a certain value

The various components are following the dynamic :

$$s_n = s_{n-1} + \epsilon_n^s \quad \text{where } \epsilon_n^s \sim \mathcal{N}(0, \sigma_{s,n}^2,]-s_{n-1}, +\infty[)$$

$$g_n^{heat} = g_{n-1}^{heat} + \epsilon_n^g \quad \text{where } \epsilon_n^g \sim \mathcal{N}(0, \sigma_{g,n}^2,]-\infty, -g_{n-1}^{heat}[)$$

$$\sigma_{s,n} = \sigma_{s,n-1} + \eta_n^s \quad \text{where } \eta_n^s \sim \mathcal{N}(0, \sigma_s^2,]-\sigma_{s,n-1}, +\infty[)$$

$$\sigma_{g,n} = \sigma_{g,n-1} + \eta_n^g \quad \text{where } \eta_n^g \sim \mathcal{N}(0, \sigma_g^2,]-\sigma_{g,n-1}, +\infty[)$$

Θ : Fixed Parameter (estimated later via PMCMC)

$$\theta = (\sigma_s, \sigma_g, u^{heat}, \kappa, \sigma)$$

Goal : Forecasting

<u>Input :</u>
Daily Temperature for a specific half-hour
(several Forecasting models are available)

<u>Ouput</u>:
Electricity Load Forecast for a specific half-hour
Time Horizon: **1 to 5 days ahead.**

- We train a different model for each half hour of the day. Here, we chose to work on 3:00PM time series.
- Therefore we cannot use the previous half-hour electricity load to predict the next one.

Method

Particle Filter algorithm

Algorithm 3.10 (Particle filter used for our application).

At time $n = 0$

1. Sample $\widehat{X}_0^j \sim \mu(x_0)$.
2. Compute $\widetilde{w}_0^j = g_0(y_0|X_0^j)$ and set $\widehat{w}_0^j \leftarrow \frac{\widetilde{w}_0^j}{\sum_{k=1}^M \widetilde{w}_0^k}$.
   - if $\widehat{\text{ESS}}(0) < 0.001M$, set $X_0^j \leftarrow \widehat{X}_0^j$ and $w_0^j \leftarrow 1/M$.
   - if $0.001M \leq \widehat{\text{ESS}}(0) < 0.5M$, use residual-multinomial resample (see Algorithm 3.5) and regularisation move (see Algorithm 3.6) steps to set X_0^j and w_0^j.
   - if $0.5M \leq \widehat{\text{ESS}}(0)$, set $X_0^j \leftarrow \widehat{X}_0^j$ and $w_0^j \leftarrow \widehat{w}_0^j$.

At time $n \geq 1$

1. Sample $\widehat{X}_n^j \sim f_n(x_n|X_{n-1}^j)$.
2. Compute $\widetilde{w}_n^j = w_{n-1}^j g_n(y_n|X_n^j)$ and set $\widehat{w}_n^j \leftarrow \frac{\widetilde{w}_n^j}{\sum_{k=1}^M \widetilde{w}_n^k}$.
   - if $\widehat{\text{ESS}}(n) < 0.001M$, set $X_n^j \leftarrow \widehat{X}_n^j$ and $w_n^j \leftarrow w_{n-1}^j$.
   - if $0.001M \leq \widehat{\text{ESS}}(n) < 0.5M$, use residual-multinomial resample (see Algorithm 3.5) and regularisation move (see Algorithm 3.6) steps to set X_n^j and w_n^j.
   - if $0.5M \leq \widehat{\text{ESS}}(n)$, set $X_n^j \leftarrow \widehat{X}_n^j$ and $w_n^j \leftarrow \widehat{w}_n^j$.

Initialization step: the algorithm is <u>very sensitive to the initial value</u> of the particles

Removal of outliers

Multinomial resampling instead

Non-systematic resampling

How to initialize the Particle Filter ?

Basic idea

With abritrary values

Problem in our model: Hard to find an appropriate order of magnitude for all components, especially the variances, due to complex hierarchy

Solution

Gibbs sampling

Idea: Simulate from the smoothed distribution until a date $\boldsymbol{n_0}$ - 1 using a Gibbs sampler and use the last simulated value for date $\boldsymbol{n_0}$ - 1 to initialize the Particle Filter

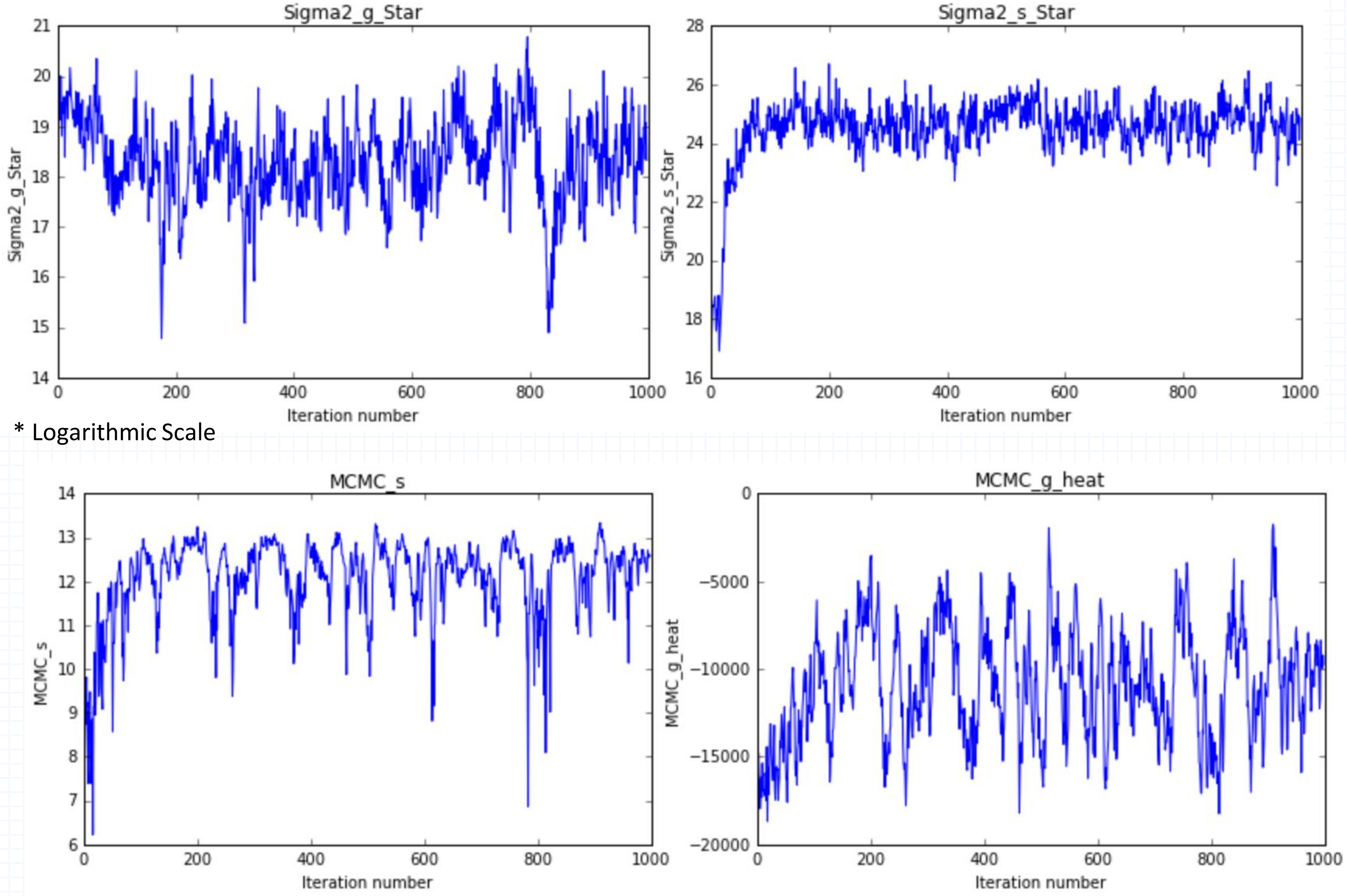
Results

Fast convergence to a right order of magnitude

14 days during a cold period (2 weeks in January), where the temperatuer are below the u^{heat} threshhold, are enough to get a good initialization and are computationally affordable!

Gibbs Convergence

1 000 iterations
Results on the last day values (n_0 - 1)



Particle Filter application

1. Parameter Estimation

The PF provides a **likelihood estimation** which can be reused in a Metropolis-Hastings algorithm in order to estimate the posterior distribution of the parameters (Marginal PMCMC algorithm)

2. Forecast

The PF provides predictions by simulating the particles of the future dates:

$$\widehat{y}_{n+\tau,i} = \mathbb{E}[x_{n+\tau}|y_{0:n,i}].$$

Parameter Estimation using PMCMC

Marginal PMCMC

From current point θ_m (and current PF estimate $L_T^N(\theta_m)$):

1. Sample $\theta_\star \sim H(\theta_m, \mathrm{d}\theta_\star)$
2. Run a PF so as to obtain $L_T^N(\theta_\star)$, an unbiased estimate of $L_T(\theta_\star) = p(y_{0:T}|\theta_\star)$.
3. With probability $1 \wedge r$, set $\theta_{m+1} = \theta_\star$, otherwise $\theta_{m+1} = \theta_m$ with

$$r = \frac{p(\theta_\star)L_T^N(\theta_\star)h(\theta_m|\theta_\star)}{p(\theta_m)L_T^N(\theta_m)h(\theta_\star|\theta_m)}$$

Goal: Estimate the parameter $\theta = (\sigma_s, \sigma_g, u^{heat}, \kappa, \sigma)$

Parameter Estimation using PMCMC - Difficulties

Difficulty 1: Proposal Choice for $\theta = (\sigma_s, \sigma_g, u^{heat}, \kappa, \sigma)$

Gaussian Random Walk ?

- Truncated Normal distribution for the 3 variance parameters => Non-symmetric proposal
- Kappa is a vector which sum must be 1 => no random walk can be used

Difficulty 2: κ estimation and proposal

Independent uniform Dirichlet proposal ?

- In practice, we tested this method and it led to a acceptance ratio close to 0
- So we fixed κ to (1/8,…, 1/8) where 8 was our number of daytypes => **Strong hypothesis**: we assumed there was no difference between daytypes

Difficulty 3: Tuning the hyperparameters

Try many different values ?

- Computational intensive algorithm : 16 hours to run 300 Metropolis iterations => we have to allow a large variance in order to explore a large range of values in a small amount of iterations

Results

Result of the PMMH Algorithm (1/2)

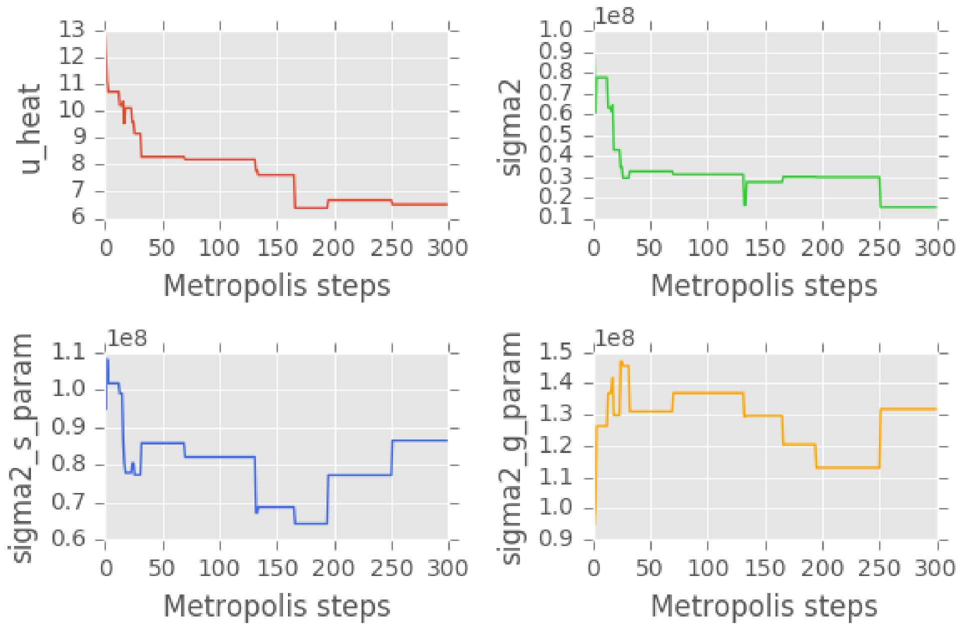
PMMH Result (with a burn out of 30 iterations)

	Acceptance rate	u^{heat}	σ^2	σ_s^2	σ_g^2
PMMH	5.35 %	7.3	27E+06	79E+06	127E+06

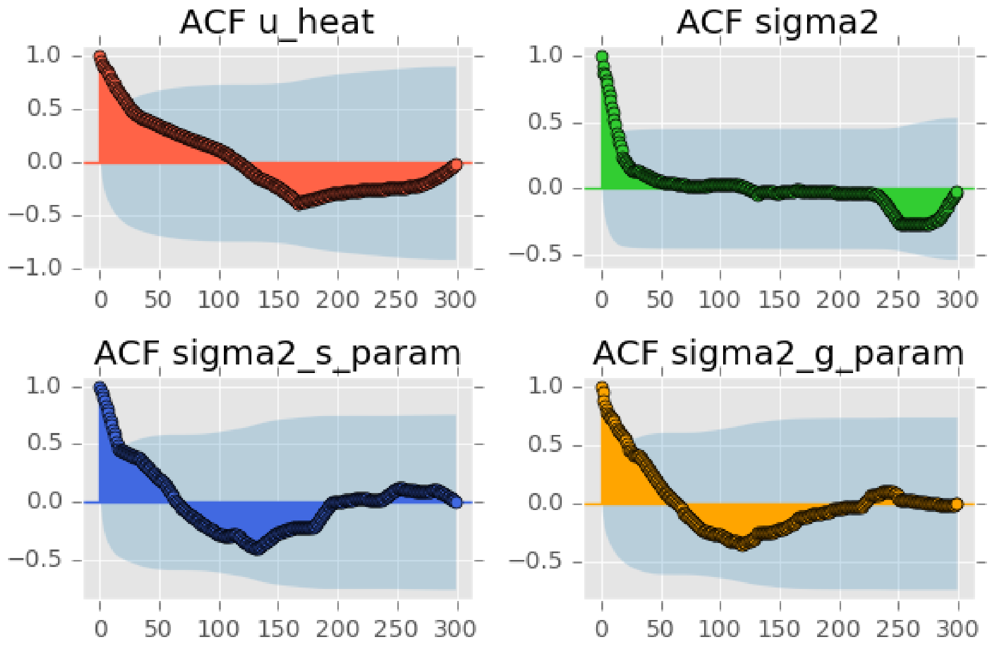
- This result has been computed using a PMMH algorithm with 300 iterations
- We observed similar results in previous versions of PMMH
- The value of u^{heat} is lower than expected (around 14 C° in the article)
- The acceptance rate is low because the model is very sensitive to the parameters → for some parameters the likelihood can be $-\infty$!

Result of the PMMH Algorithm (2/2)

Convergence of the PMMH



Evolution of the parameters



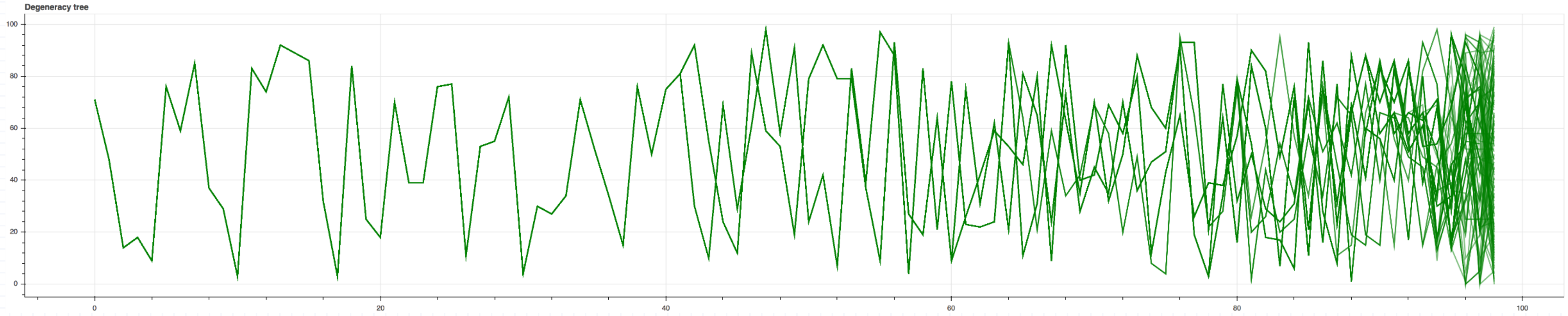
ACF

Degeneracy in the Particle Filter

- A particle can move too far from the real value → after some iterations almost all particles are too far from the real value : the model is degenerate

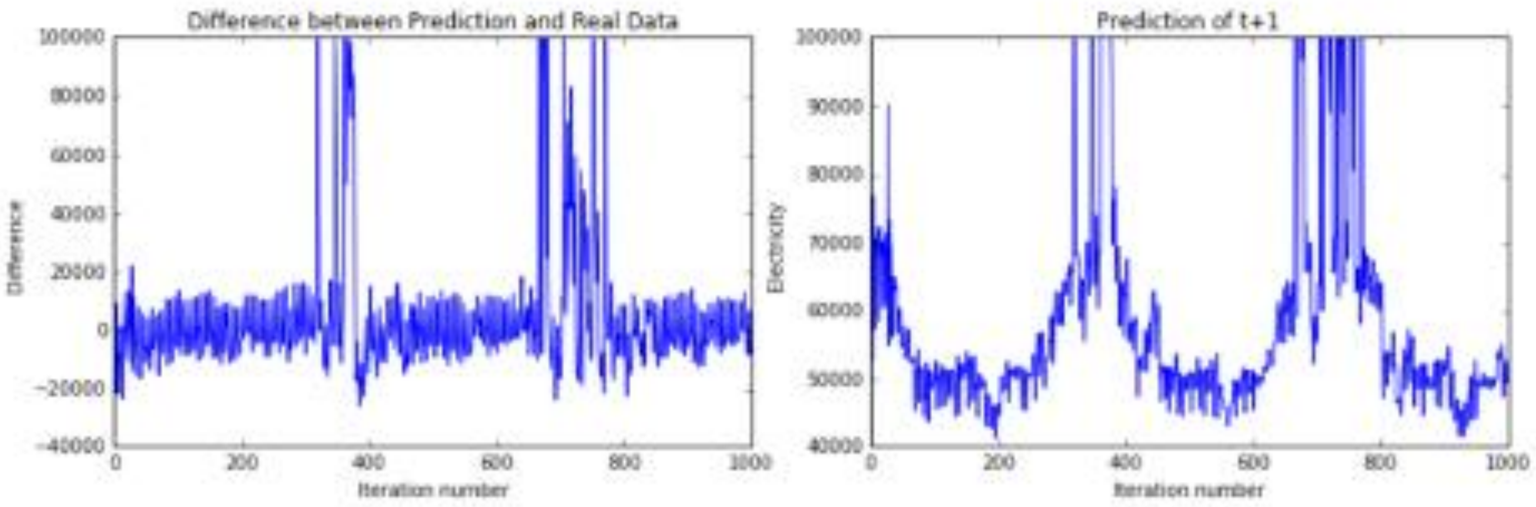
- When the model is too degenerate, we resample the particle → we can observe the genealogy of the particles.

Genealogy of the particles



Forecast Result (1/2)

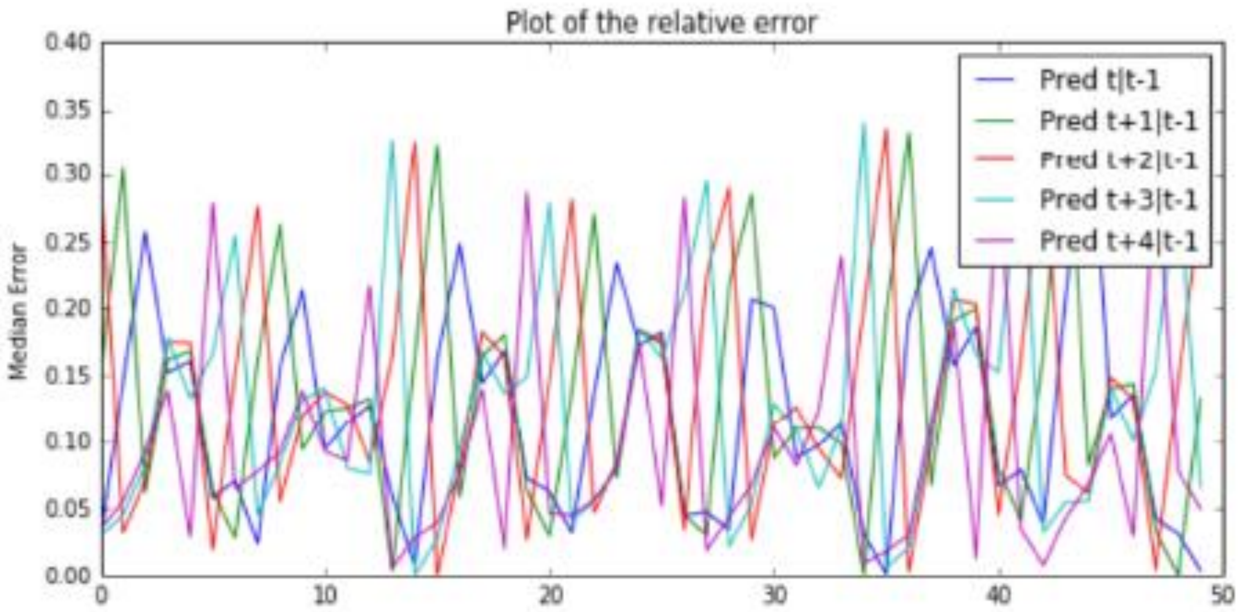
Prediction of the Electricity Loads

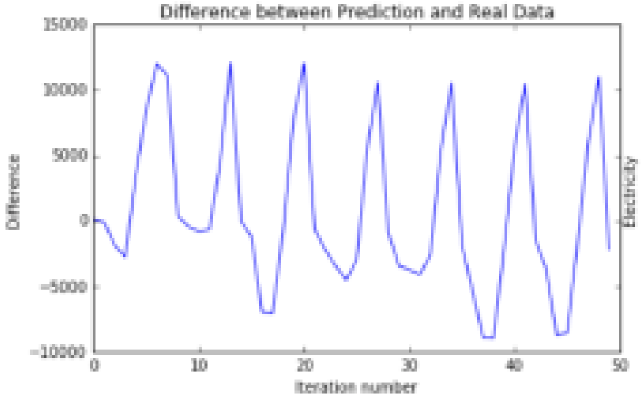


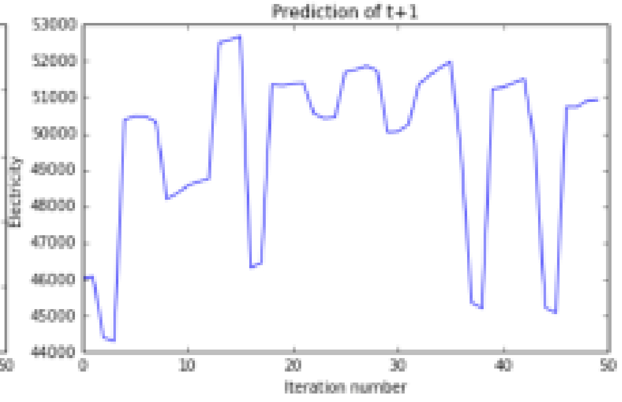
- We can observe an annual seasonality
- We have big outliers during winter, when the temperature drops below u^{heat}
- The MAPE is around 10% without these outliers (20% with).

Forecast Result (2/2)

Forecasting during 50 days







Conclusion

Practical Issues

- All the parameters have very strong links together, and results are based on random simulation
  → it is difficult to explain why we sometimes generate extreme values

- The different calculus can provide very big result (displayed as Inf in Python) or very close to zero (displayed as zero) → we had to use logarithmic transformations

- The different algorithms take a lot of times to run :
  - ✓ Around 10 minutes for a PF with 1000 particles and 1000 iterations
  - ✓ 16 hours for the PMMH with 300 iterations

Further Improvements

- Increase the length of the different algorithms / the number of particles for a better accuracy

- Build a model for every half-hour → we only forecast the electricity load at 3:00pm

- Compare our model with other models (like SARIMA, Kalman Filters, Rule-Based Modelling (Dordonnat & al, 2008)…).

Conclusion