Deep Neural Networks for Irregular Atronomical Time Series

Pavlos Protopapas
Institute for Applied Computational Science, Harvard
Collaborators: Abhishek Malali and Reinier Maat

Outline

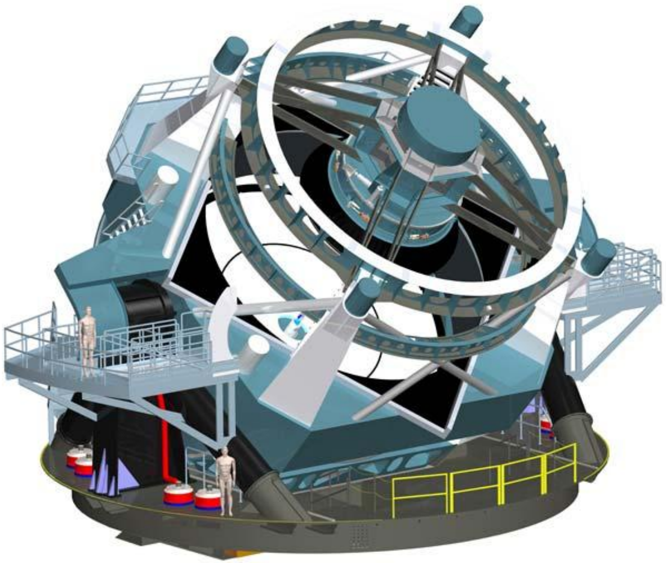
- Motivation
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- Autocorrelation
  - Regular
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

Outline

- **Motivation**
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- Autocorrelation
  - Regular
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

Motivation

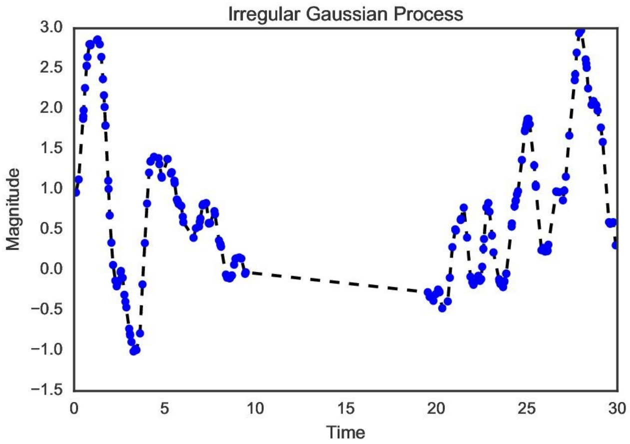
- Anticipated explosion in astronomical data with expected commissioning of the Large Synoptic Survey Telescope (LSST).
- Requirement of an early warning system to predict future events of interest.
- Increases chances of observing rare events and improving the allocation of the resources in astronomy.



http://ast.noao.edu/facilities/future/lsst

Problem Statement

- Time series datasets in astronomy are irregular in nature.
- Irregular time series are also found in transactional data and climatology.
- Conversion to regular time series; applying predictive models like ARIMA, Kalman filters and State Space Analysis.
- Model time series as non-linear models and solve the prediction problem in the irregular time domain.



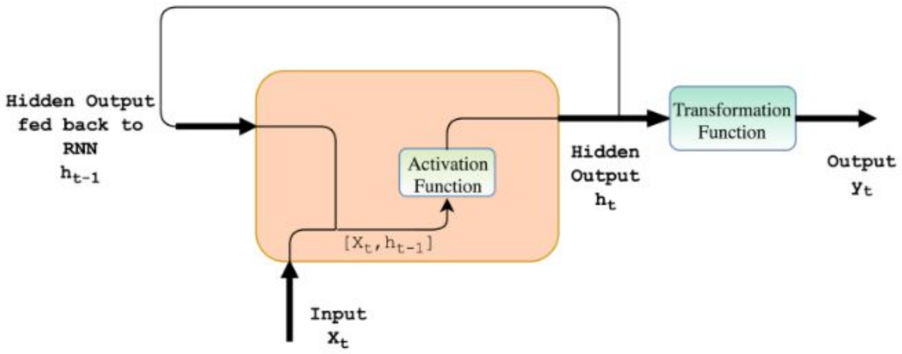
Example of an Irregular and noisy time series.

Outline

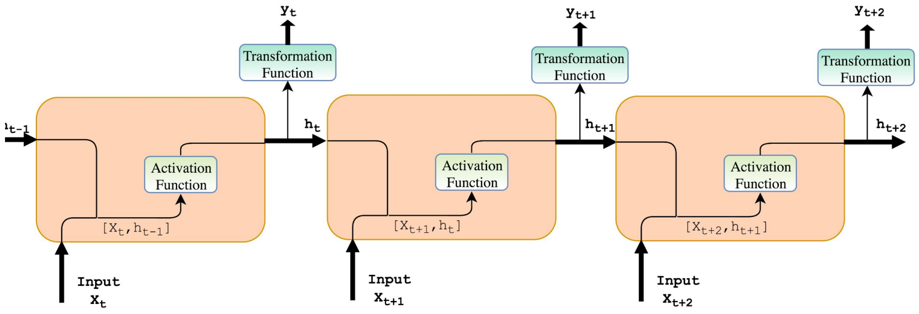
- Motivation
- **Recurrent Neural Networks**
  - LSTM
  - Echo State Networks
- Autocorrelation
  - Regular
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

Recurrent Neural Networks

- Neural network architecture with recurrent edges spanning time steps.
- Suffers from vanishing gradient problem.
- Limited ability to learn long term dependencies.



Single recurrent network unit

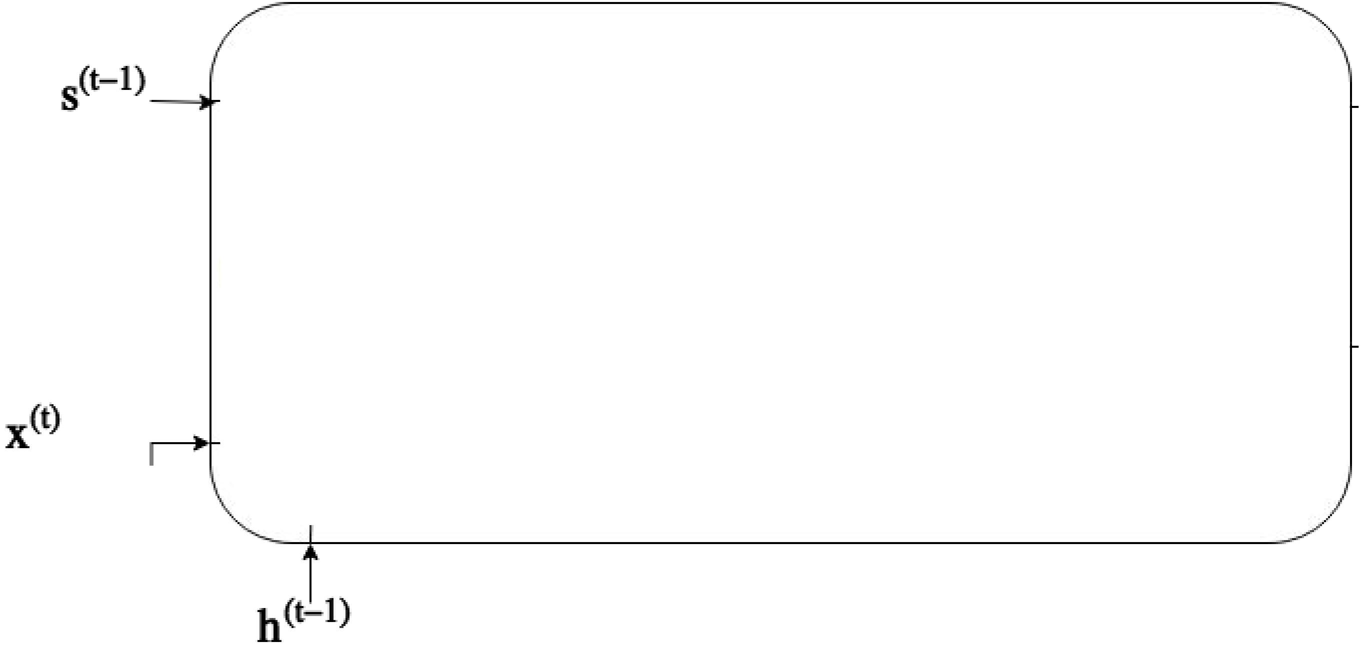


Recurrent network unrolled over time

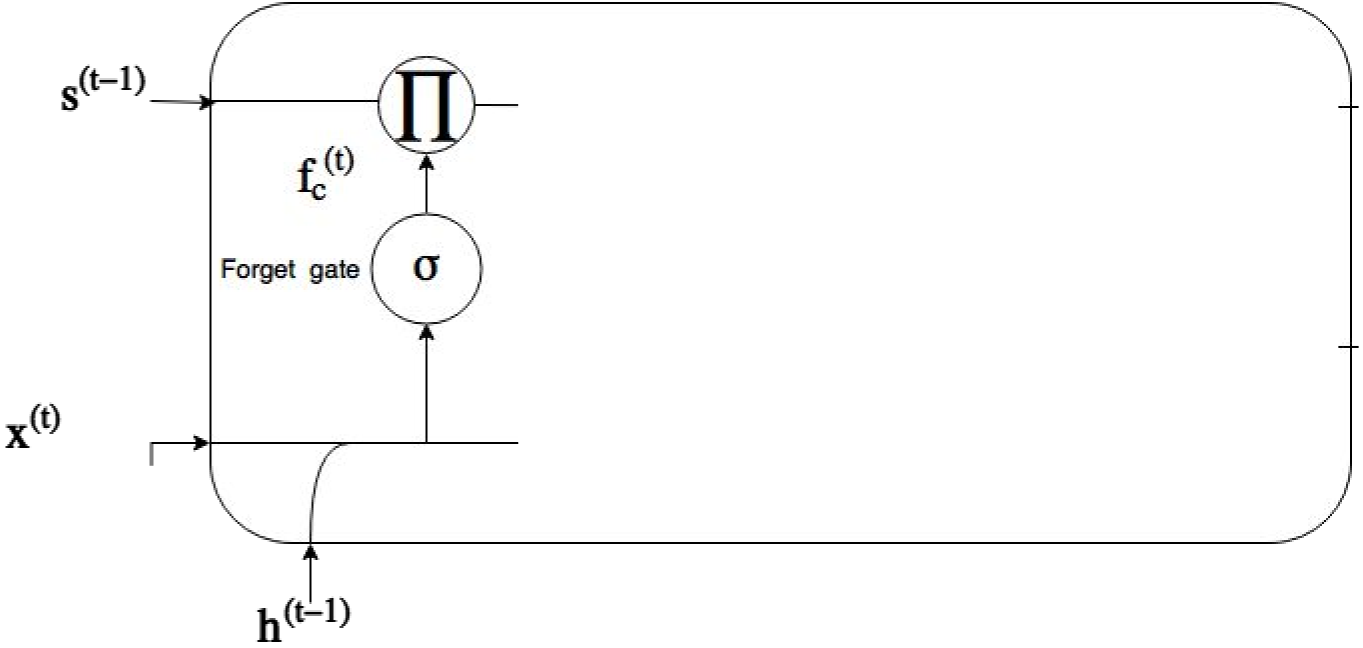
Outline

- Motivation
- Recurrent Neural Networks
  - **LSTM**
  - Echo State Networks
- Autocorrelation
  - Regular
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

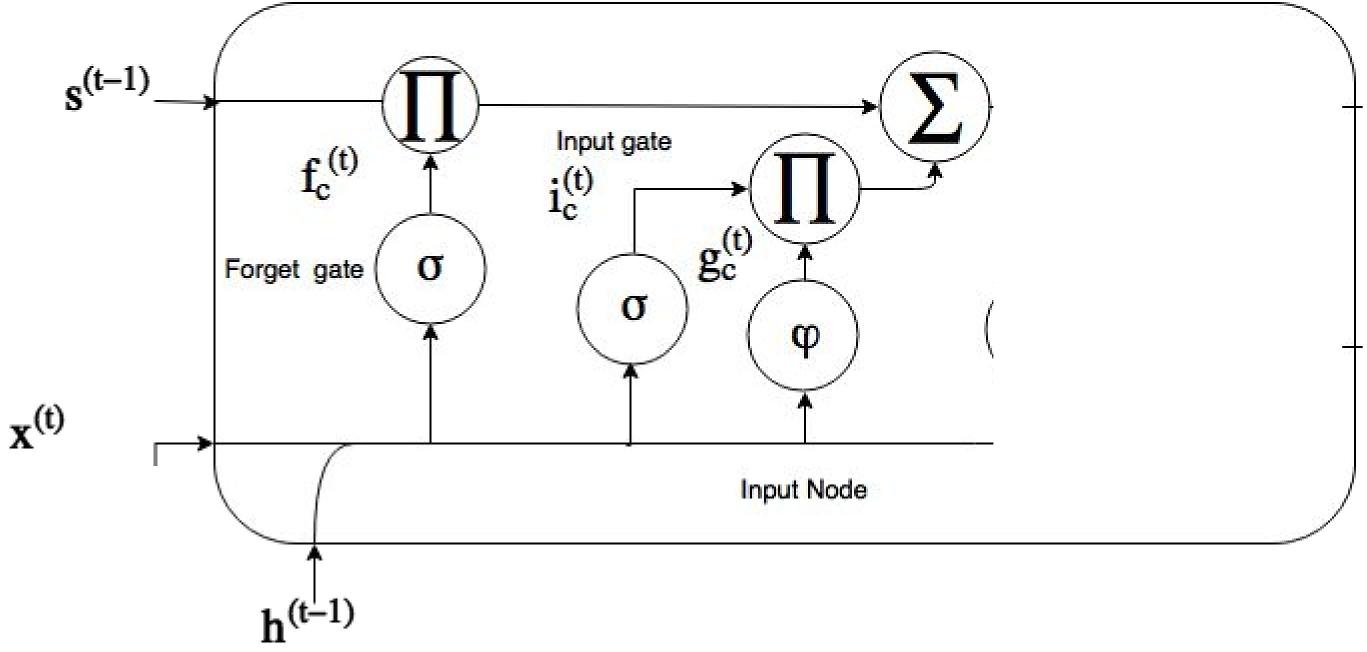
Long Short Term Memory



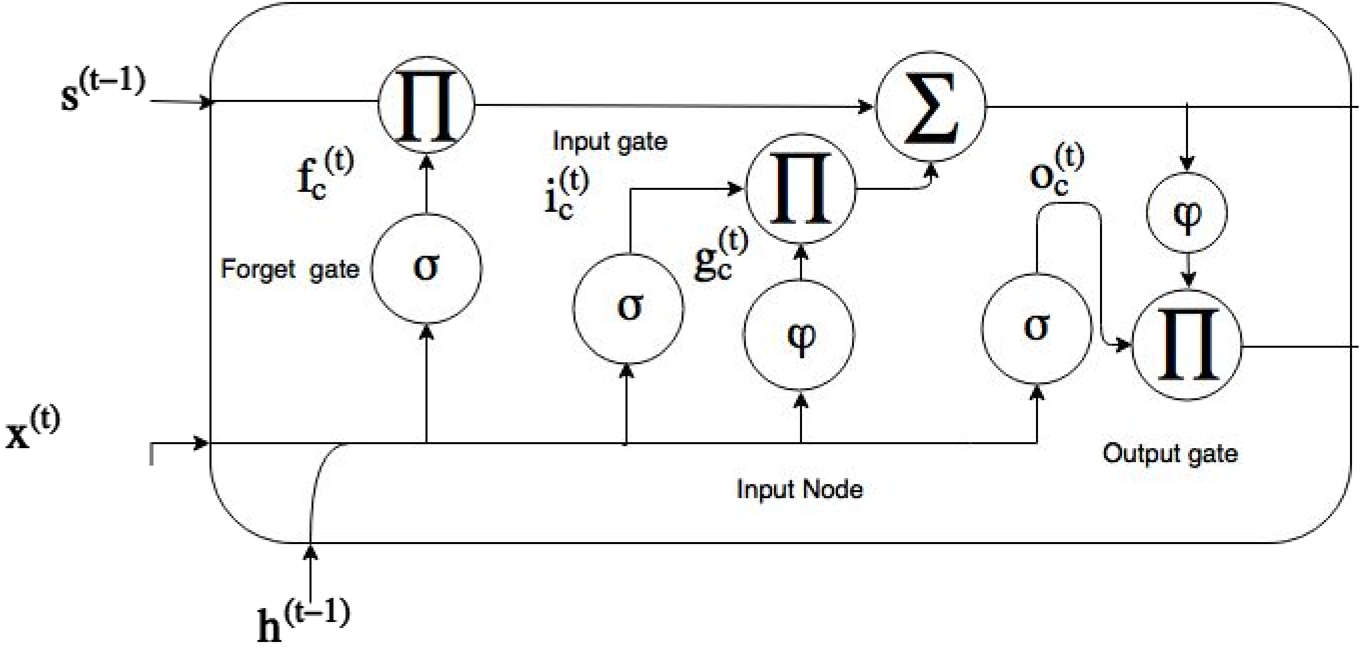
Long Short Term Memory



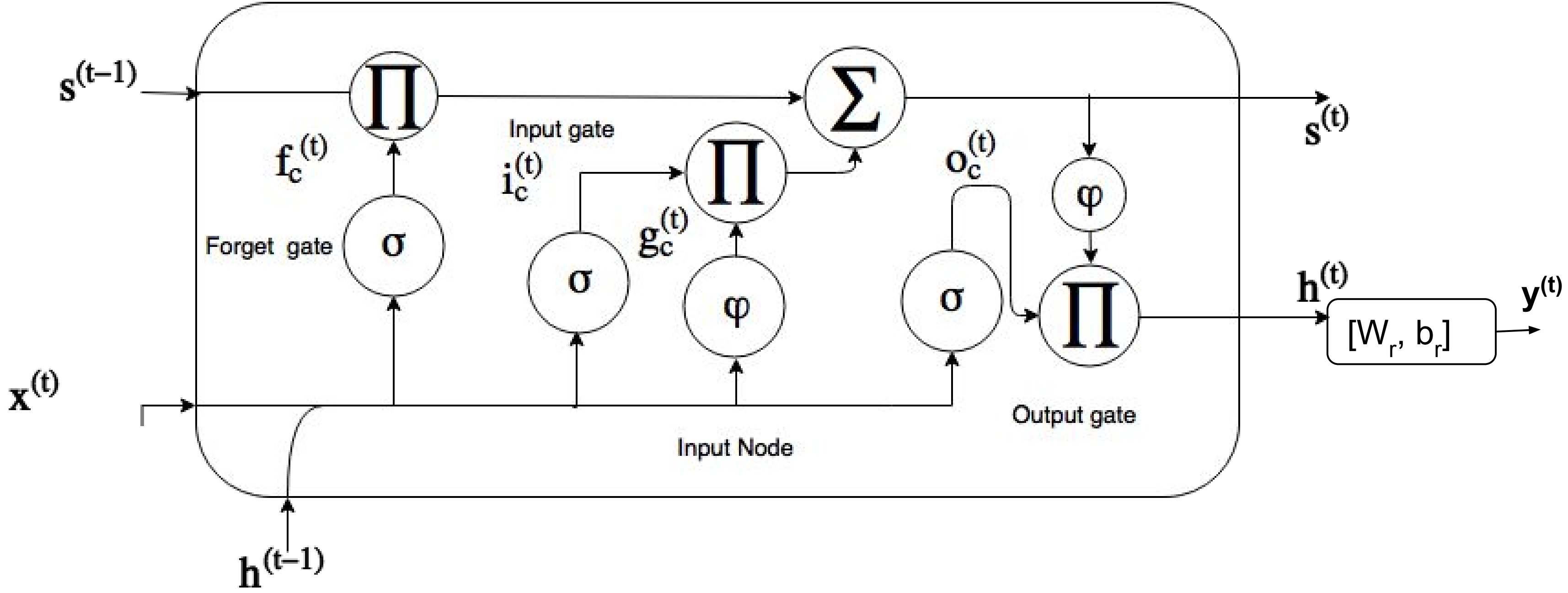
Long Short Term Memory



Long Short Term Memory



Long Short Term Memory



Outline

- Motivation
- Recurrent Neural Networks
  - LSTM
  - **Echo State Networks**
- Autocorrelation
  - Regular
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

Echo State Networks

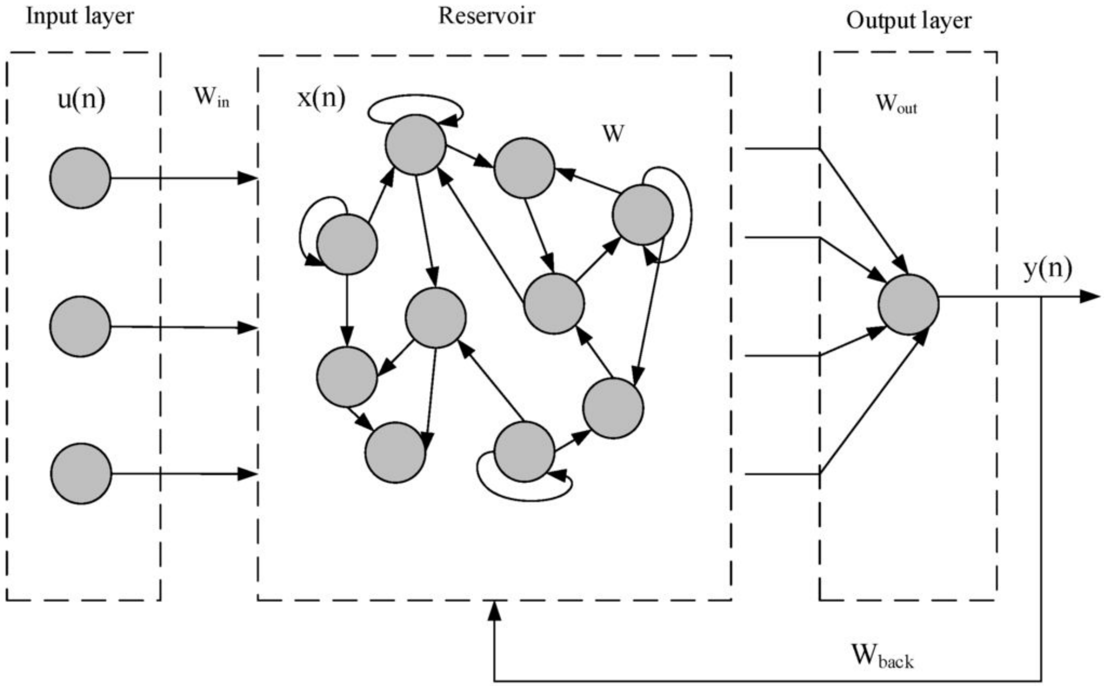


Figure 1: Example in ESN

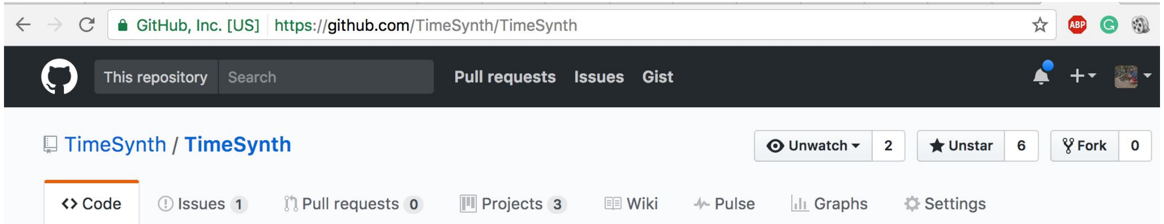
Input Features

Data is standardized initially and processed. To generate the prediction at time t_j to generate the following inputs:

- Magnitude at t_{j-1}
- Time Difference $t_j - t_{j-1}$
- Velocity at t_{j-1}
- Acceleration at t_{j-1}

Data

- Synthetic datasets - Implemented in TimeSynth
  - Regular time series
  - Irregular time series
    - Time stamps generated as a mixture of regular time stamps with gaussian perturbations
    - Time stamps generated from a uniform distribution



- Astronomy Datasets
  - HiTS - High Cadence Transient Survey
  - MACHO - Massive Astrophysical Compact Halo Objects Survey

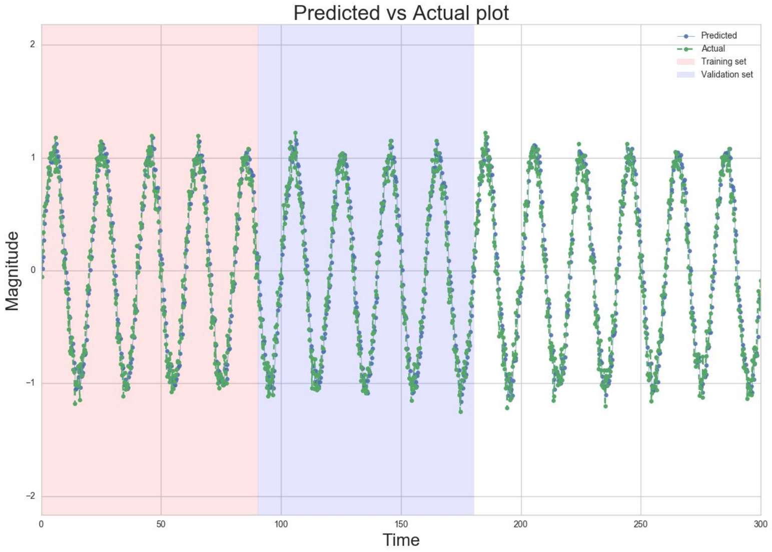
Outline

- Motivation
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- **Autocorrelation**
  - Regular
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

Correlations in Residuals

LSTM with input features on an irregular time series with 1000 samples, f = 0.05Hz and added noise of 0.1 standard deviation

$$\mathcal{L} = \sum_{j}^{N} \frac{(\hat{y}_j - y_j)^2}{N}$$



Outline

- Motivation
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- Autocorrelation
  - **Regular**
  - Irregular
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

Autocorrelation for Regular Time Series

Autocorrelation is a measure of non-randomness in data and identify an appropriate time series model if data is not random.

Autocorrelation for lag k,

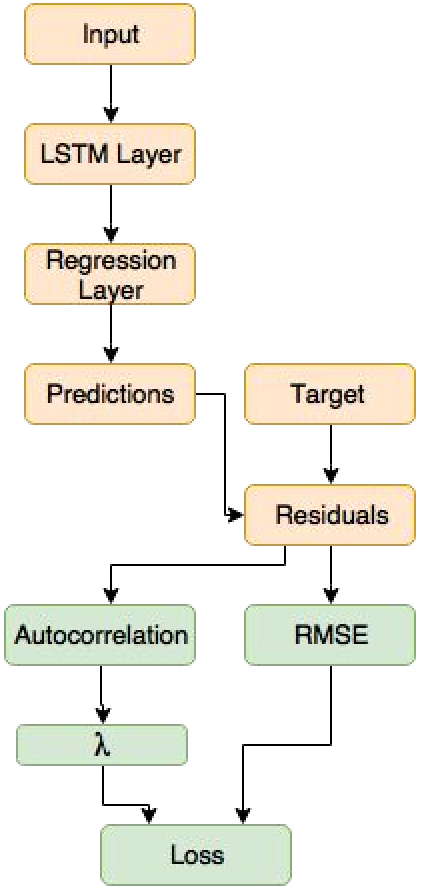
$$\rho(k) = \frac{\sum_{i=1}^{N-k}(y_i - \bar{y})(y_{i+k} - \bar{y})}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

Experiments on Regular Time Series

- 1st lag autocorrelation $\rho(1)$ of the residuals is computed
- Autocorrelation of the residuals is included as a part of the loss function
- Modified loss function

$$\mathcal{L} = \sum_{j}^{N} \frac{(\hat{y}_j - y_j)^2}{N} + \lambda \rho^2$$

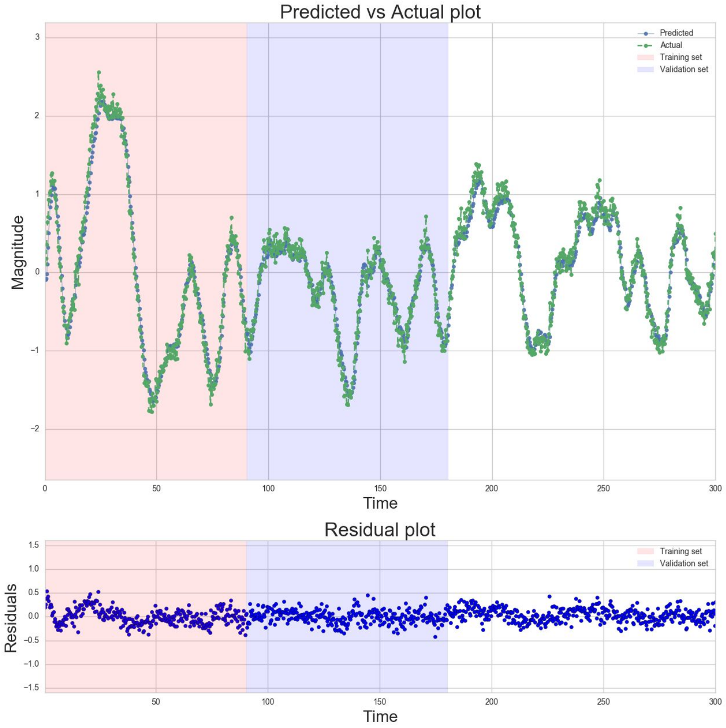
where λ is the regularization parameter

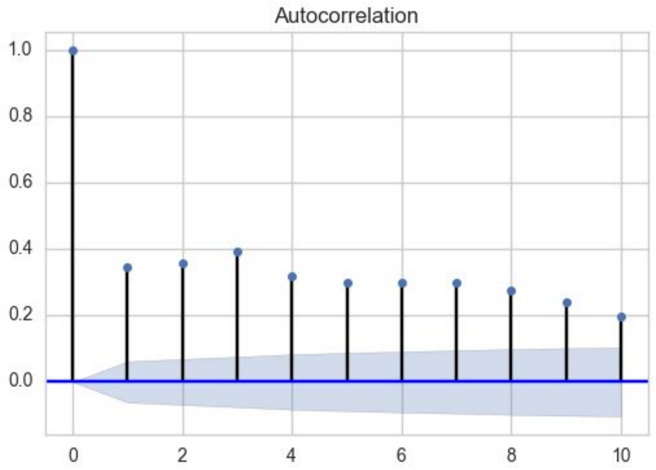


Network Architecture for Regular time series

Experiments on Regular time Series

Without regularization



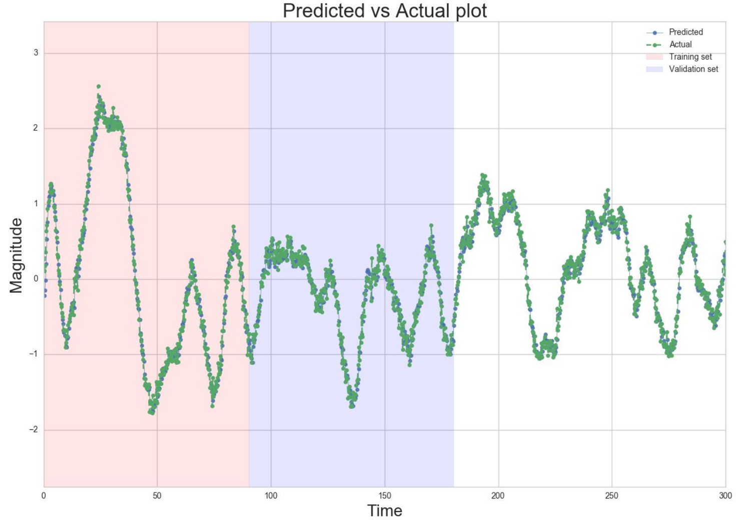


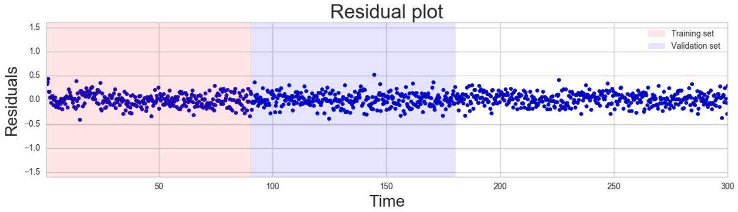
Regularization parameter λ = 0

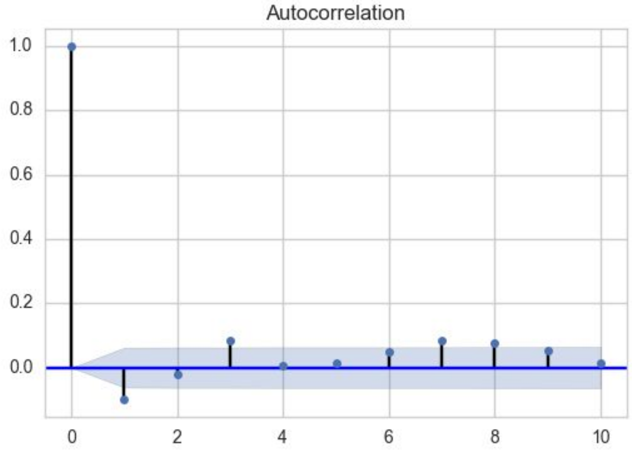
Results and autocorrelation plots for a gaussian process with SE kernel(σ = 0.5, L= 4) and added noise of standard deviation 0.1

Experiments on Regular time Series

With regularization







Regularization parameter λ = 1

Results and autocorrelation plots for a gaussian process with SE kernel(σ = 0.5, L= 4) and added noise of standard deviation 0.1

Outline

- Motivation
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- Autocorrelation
  - Regular
  - **Irregular**
- Noisy Data
- Echo State Networks
  - Hyper-parameters optimization

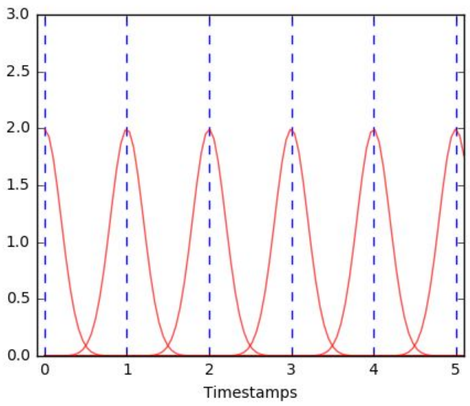
Autocorrelation for Irregular Time Series

Gaussian Perturbations

Distribution of the i^th time stamp

$$t_i = \mathcal{N}(ri, r\sigma_p)$$

Where *r* is the resolution and σ_p is the gaussian standard deviation.



Gaussian Perturbation Timestamp Sampling

Time difference distribution

$$\Delta t \sim t_{i+1} - t_i \sim \mathcal{N}(r, \sqrt{2} r\sigma_p)$$

$$\Delta t = 1 + \Delta t_n$$

Where $\Delta t_n \sim \mathcal{N}(0, \sqrt{2}\sigma_p)$

Autocorrelation for Irregular Time Series

Gaussian Perturbations

Continuous time autoregressive equation for Irregular time series residuals

$$r_{t_{i+1}} = \phi^{t_{i+1}-ti} r_{t_i} + \epsilon$$

Where $\epsilon \sim \mathcal{N}(0, \sigma)$ and $\Phi^{1 + \Delta tn}$ is equivalent to *ρ(1)*.

$$r_{t_{i+1}} = \phi^{1+\Delta t_n} r_{t_i} + \epsilon$$

$$E[\phi^{1+\Delta t_n}] = \phi\left[1 + \sigma_p^2 \ln(\phi)^2 + \frac{(\sigma_p^2 \ln(\phi)^2)^2}{2} + ...\right]$$

$$E[\phi^{1+\Delta t_n}] \simeq \phi$$ For small values of σ_p

Autocorrelation for Irregular Time Series

Uniform Time Sampling

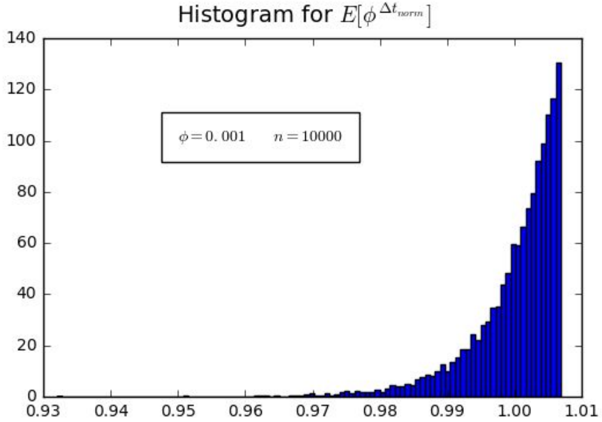
Samples [U_1, U_2,...,U_n] are drawn from an uniform distribution U[0,T] and ordered into [$U_{(1)}$, $U_{(2)}$, ..., $U_{(n)}$] where $U_{(k)}$ is the kth order statistic.

$$\Delta t \sim \frac{1}{T}\beta(1,n)$$

Where n is the number of samples and β stands for Beta Distribution.

$$r_{t_{i+1}} = \phi^{1+\frac{\Delta t - \bar{\Delta} t}{\bar{\Delta} t}} r_{t_i} + \epsilon$$

$$1 - \epsilon < E[\ \phi^{\frac{\Delta t - \bar{\Delta} t}{\bar{\Delta} t}}\ ] < 1 + \epsilon$$



Autocorrelation for Irregular Time Series

Estimating the autocorrelation from data

Continuous Autoregressive equation

$$r_{t_{i+1}} = \phi^{1+\frac{\Delta t - \bar{\Delta} t}{\bar{\Delta} t}} r_{t_i} + \epsilon$$

$$\Delta t_n \simeq \frac{\Delta t - \bar{\Delta} t}{\bar{\Delta} t}$$

Log-Likelihood

$$\ln(\mathcal{L}) \propto \sum_j \ln(\nu) + \sum_j \frac{e^2}{2\nu^2}$$

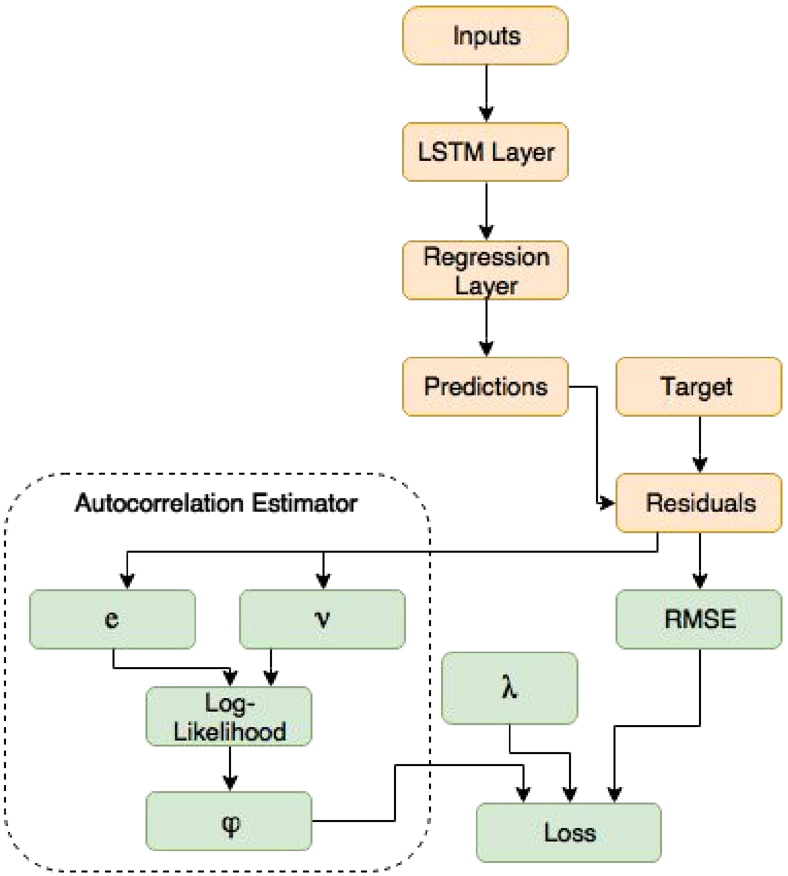
$$\nu = \sigma \qquad d_j = \Delta t_{n_j}$$

$$e_j = \phi^{1+d_j} r_{t_j} - r_{t_{j+1}}$$

Log-Likelihood is minimized by SGD until the exact parameter of Φ and σ are estimated. Heuristics used alongside of Log Likelihood for improved stability.

Network Architecture

$$\mathcal{L} = \sqrt{\frac{\sum_i^N (\hat{y} - y)^2}{n} + \lambda|\phi|^2}$$



Network Architecture for Irregular time series

Outline

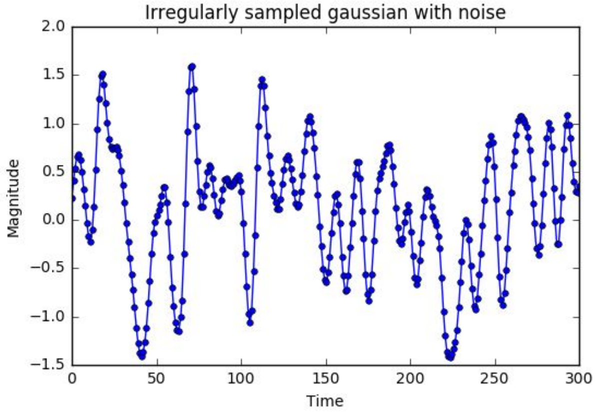
- Motivation
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- Autocorrelation
  - Regular
  - Irregular
- **Noisy Data**
- Echo State Networks
  - Hyper-parameters optimization

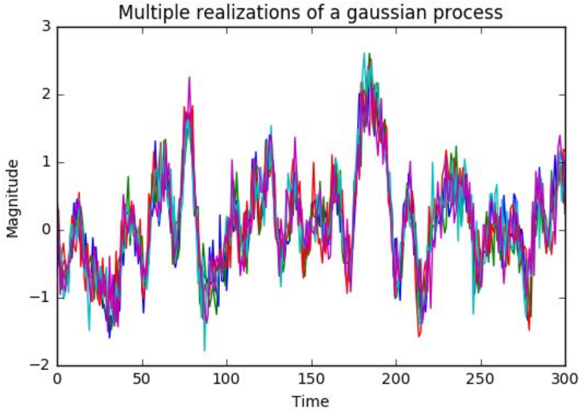
Multiple Error Realizations

Error budget is available for every measurement in astronomical light curves.

Using the error data, new realizations for the data can be generated for training.

Results indicate that using multiple realizations of the data helps the model become noise invariant.





Outline

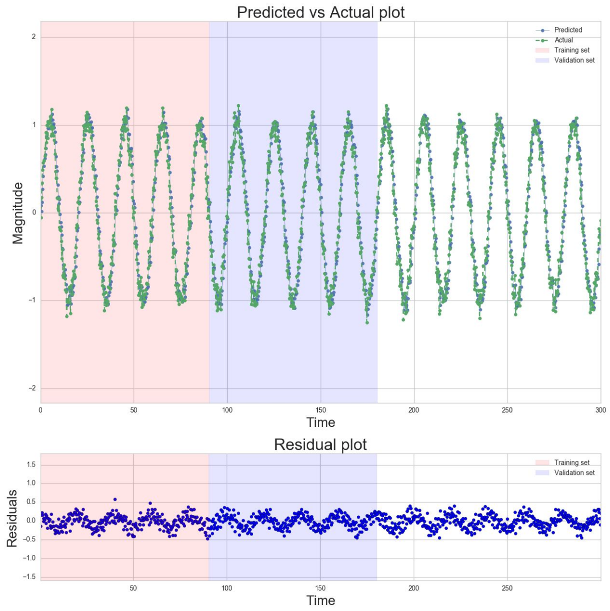
- Motivation
- Recurrent Neural Networks
    - LSTM
    - Echo State Networks
- Autocorrelation
    - Regular
    - Irregular
- Noisy Data
- Echo State Networks
    - Hyper-parameters optimization

Results

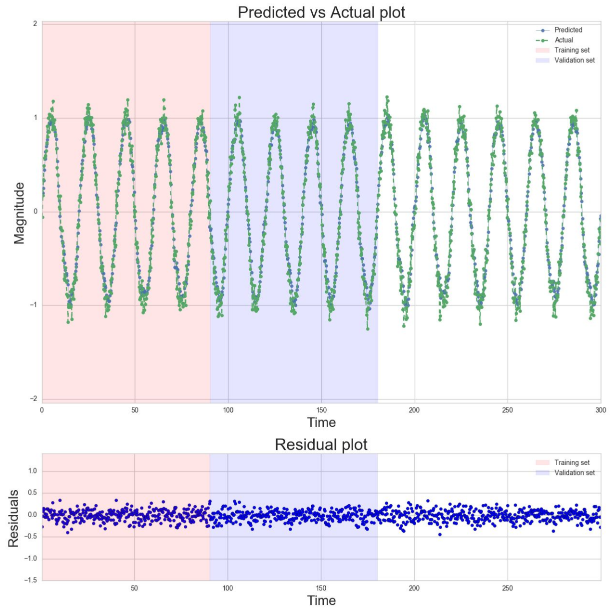
Synthetic Time Series - Gaussian Perturbations

Dataset with 1000 samples, Sinusoidal function f = 0.05Hz, T=300s, noise=0.1, σ_p = 0.2

Regularizer λ = 0



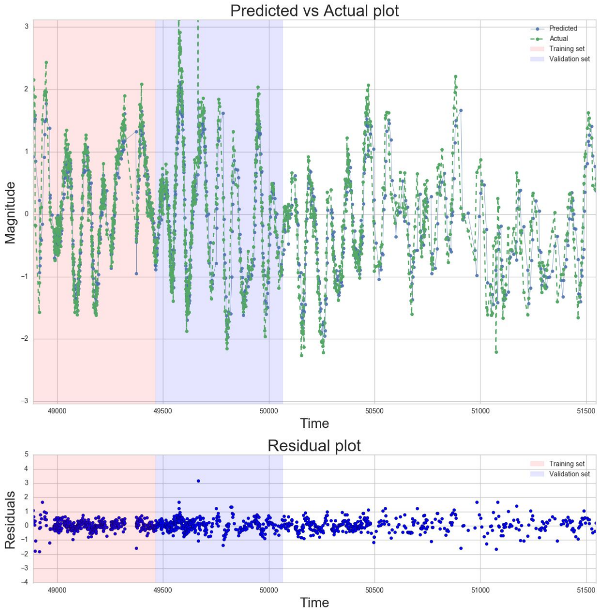
Regularizer λ = 50



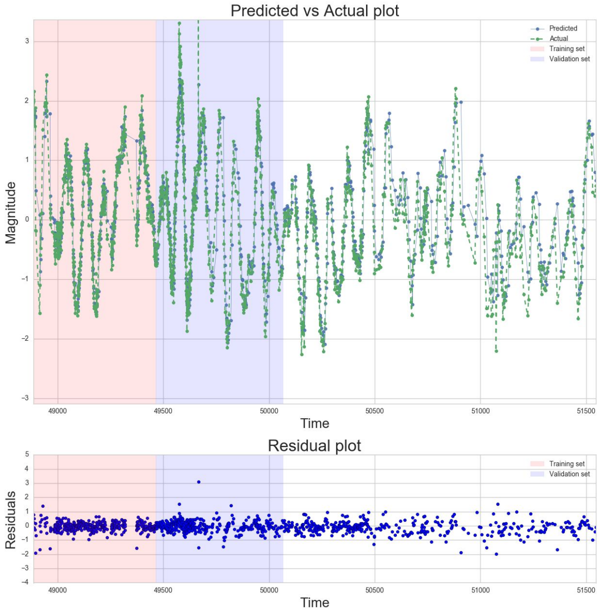
Results

Long Period Variables(with multiple realizations training)[LC-1-3319]

Regularizer λ = 0



Regularizer λ = 10



Results

Long Period Variables

Table comparing different training methods for LC-1-3319

Metric	λ = 0	λ = 1	λ = 10	λ = 0(with MR)	ARIMA(5,1,0)*
Training RMSE	0.115	0.119	0.118	0.123	0.277
Validation RMSE	0.179	0.154	0.156	0.160	0.296
Testing RMSE	0.227	0.197	0.212	0.216	0.301
Training R^2	0.815	0.842	0.838	0.830	0.251
Validation R^2	0.819	0.870	0.865	0.858	0.223
Testing R^2	0.646	0.728	0.709	0.679	0.198
Autocorrelation	0.311	0.080	0.072	0.121	0.064
Residual Noise	0.364	0.332	0.338	0.350	-

Outline

- Motivation
- Recurrent Neural Networks
  - LSTM
  - Echo State Networks
- Autocorrelation
  - Regular
  - Irregular
- Noisy Data
- **Echo State Networks**
  - Hyper-parameters optimization

Echo State Networks

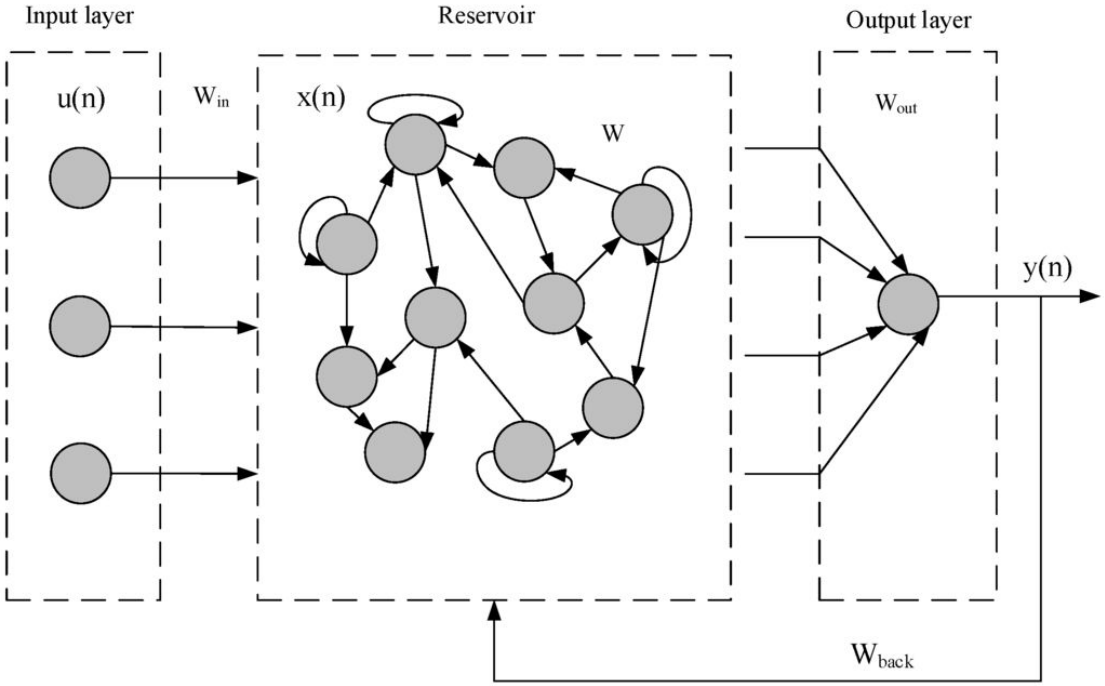
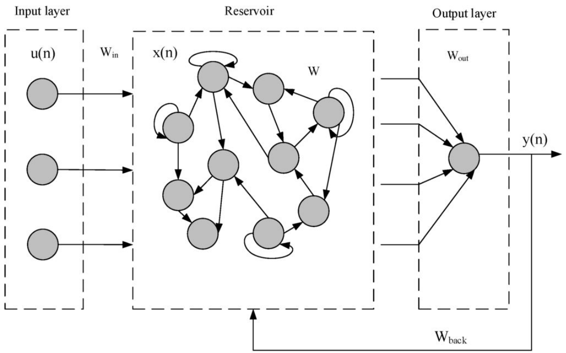


Figure 1: Example in ESN

Echo State Networks

- Recurrent Neural Net
- Reservoir (= state and transition weights)
  - Acts as nonlinear transformation and memory
- Randomly initialized adjacency matrix (many nodes)
- No backpropagation → No weight updates
  - Faster training
- Learn *only* the out-weights (called readouts)
  - E.g. through Ridge Regression



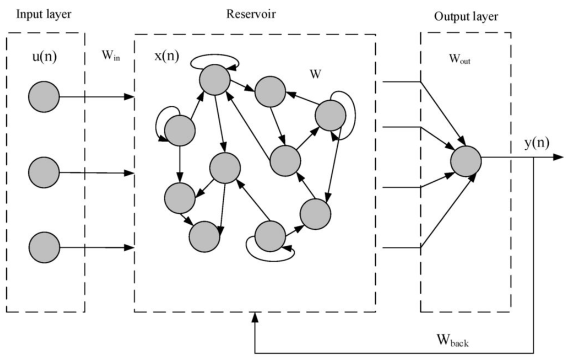
Echo State Networks

- Equations:
  - Generate state matrix **X** iteratively:

$$\mathbf{x}(t) = \tanh\left(\mathbf{W}_{in}\mathbf{u}(t) + \mathbf{W}\mathbf{x}(t-1)\right)$$

  - Train readout weights $\mathbf{W}_{\mathbf{out}}$ from state matrix **X**:

$$\mathbf{W}_{\mathbf{out}} = ((\mathbf{X}^T\mathbf{X})^{-1} + \lambda\mathbf{I})\mathbf{X}^T\mathbf{Y}$$



- Reservoir topology can be chosen
  - ‘Vanilla’ ESN: random weight matrix **W**
  - Simple Cyclic Reservoir (SCR):
    - Weights **W** arranged cyclic
  - Cyclic Reservoir with Jumps (CRJ):
    - Like SCR, but with additional node to node connections

Hyperparameter Optimization

- Tuning essential for predictive performance
- How to choose them wisely?
  - Grid search is expensive
  - Random search inefficient (especially for ‘vanilla’ ESN)
  - Not all gradients defined, so Gradient descent is unfeasible
- Solution: Bayesian Optimization

➔ Hyperparameters in ‘vanilla’ ESN (7):
- Number of nodes
- Connectivity of nodes
- Input scaling
- Feedback scaling
- Spectral radius of Reservoir
- Leaking rate
- Regularization parameter

➔ Hyperparameters in SCR (4):
- Number of nodes
- Input weight
- Cyclic weight
- Regularization parameter

Bayesian Optimization

- Global optimization technique
  - Treats error as (unknown) function of hyperparameters
  - Gaussian Process as prior over unknown function
  - Kernel defines assumed local covariance (e.g. Matérn 5/2)
  - Lengthscale per dimension set by MAP

- Iteratively:
  - Sample next set of hyperparameters in area with most merit (utility) (e.g. Expected Improvement)

- Sample until convergence (or stop early)

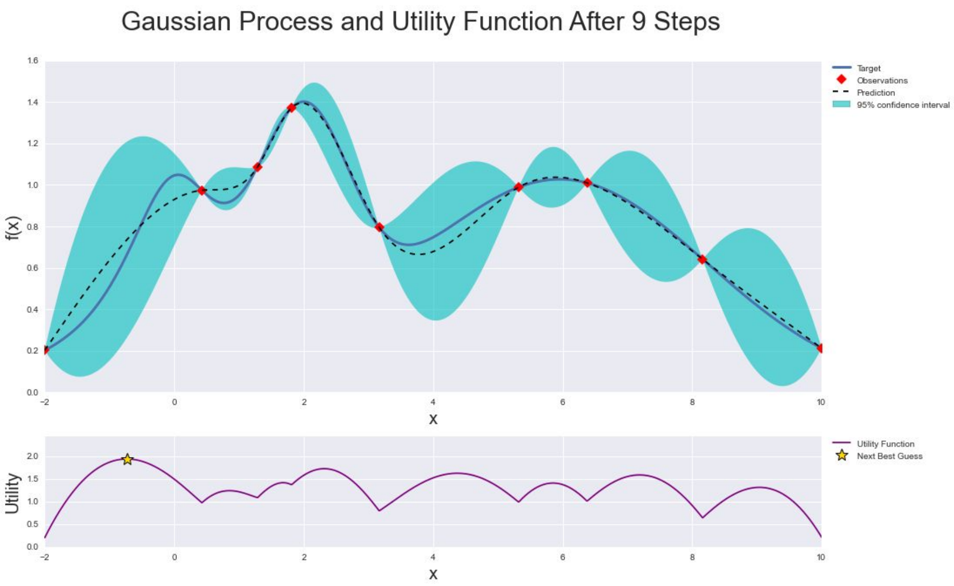


Figure 3: Example in 1 dimension

Gaussian Process and Utility Function After 9 Steps

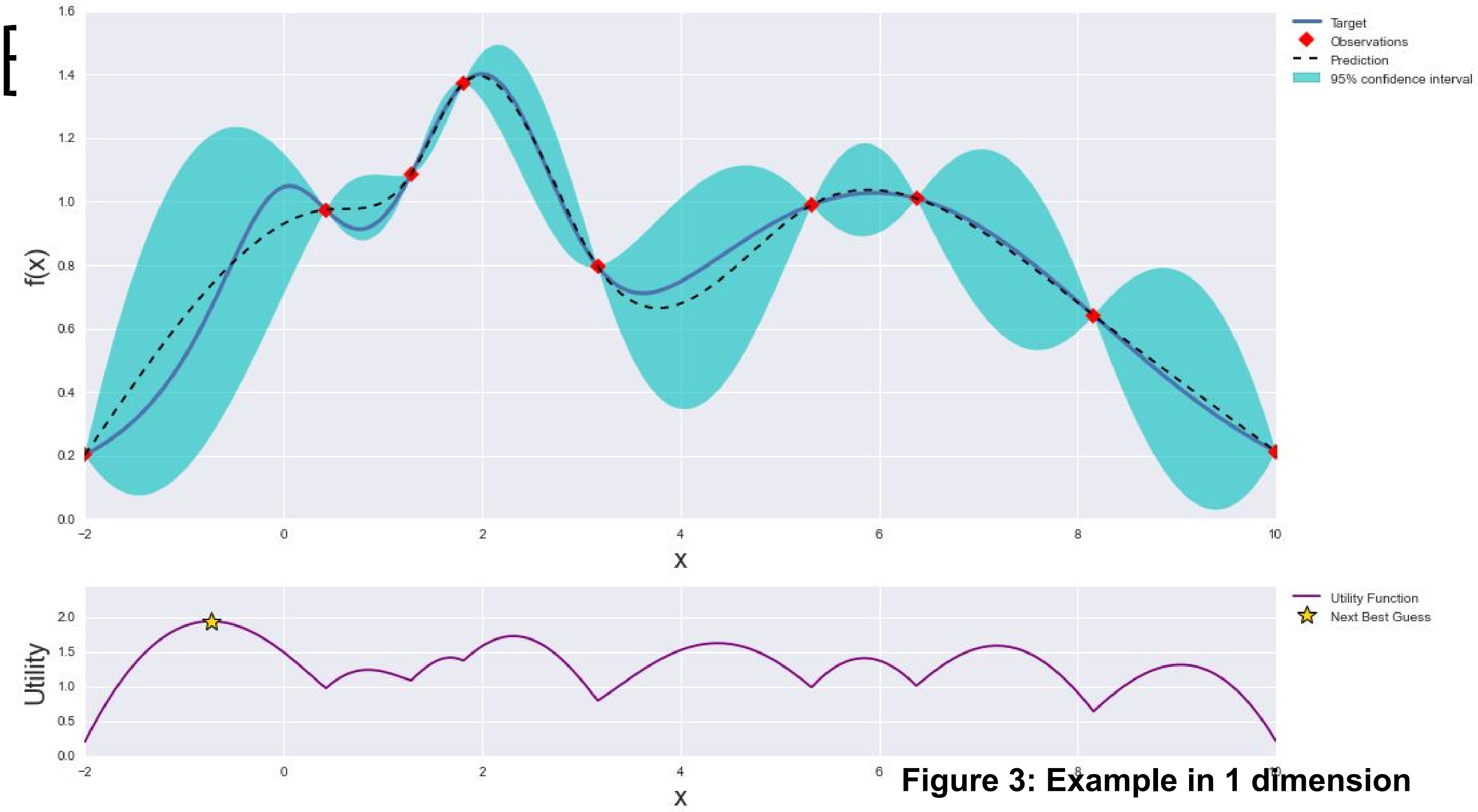


Figure 3: Example in 1 dimension

Performance: BO vs. Grid Search

- Benchmark series: NARMA 10-th order system

- BO performs better
  - Sample efficiency:
    Lower error on <1000 evaluations
  - Discrete grid vs. continuous BO

- Some overhead in modeling GP
  - Less relevant when optimizing for a collection of time series

- Eventually, performance converges with grid search

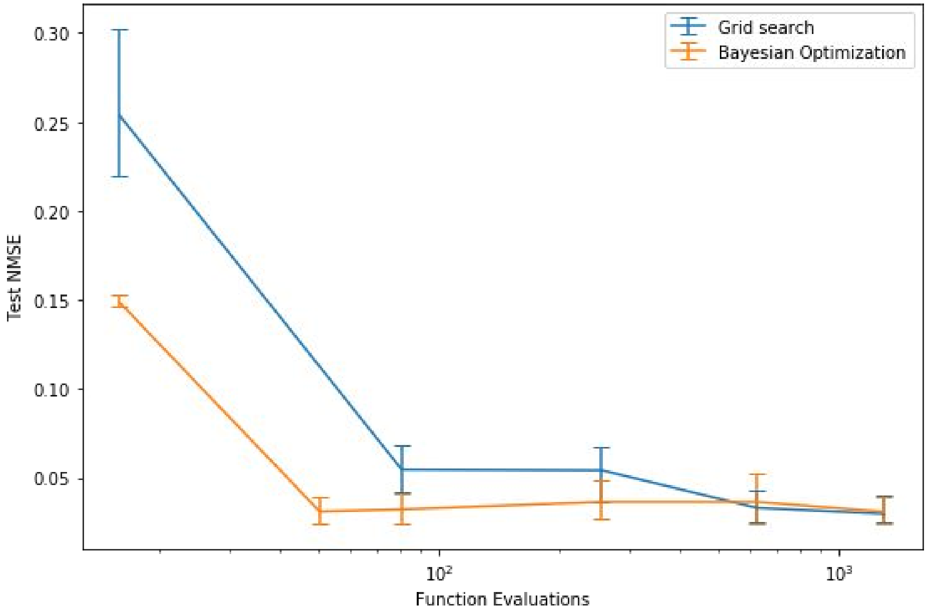


Figure 4: Performance of Grid search vs. BO

Application to Prediction

- Benchmark series: NARMA 10th-order system
- 2x improvement in NMSE

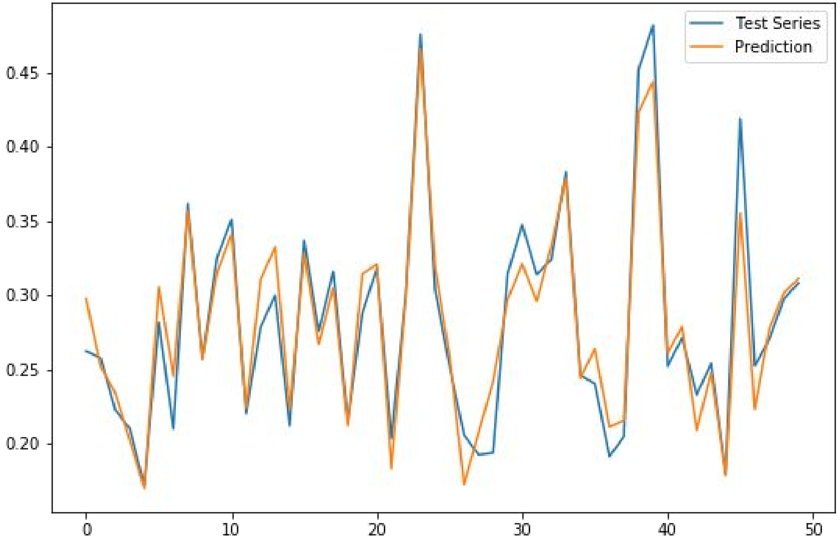


Figure 5: Step ahead prediction for grid-optimized reservoir (256 function evaluations, NMSE = 0.061)

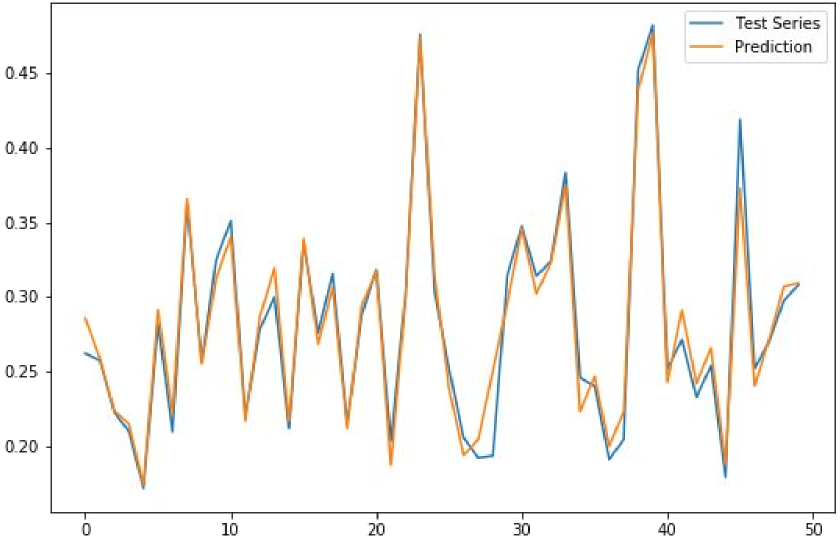


Figure 6: Step ahead prediction for BO-optimized Reservoir (256 function evaluations, NMSE = 0.031)

Application to Clustering

- Fuzzy clustering application to time series
  - Every cluster has its own model (ESN) with distinct hyperparameters

- Iteratively, until convergence:
  - Compute prediction NMSE for every series per cluster
  - Assign cluster membership inversely proportional to NMSE
  - Re-compute model and hyperparameters per cluster using Bayesian Optimization

- Outcomes:
  - Series clustered by similar dynamics, into *k* distinct clusters
  - *k* Models with hyperparameters that represent cluster well and can be used for prediction of individual series in that cluster

Additional Work

- Closed form solution to the CAR equation
- Complete the prediction model for Echo State Network
- Extend to classification for the LSTM and ESN

Conclusion

- Implemented LSTM with forget gates for irregular time series applications in TensorFlow (TimeFlow).
- Time series synthesizer for regular and irregular time series (TimeSynth).
- Estimating autocorrelation for irregularly spaced residuals in data.
- Modified LSTM loss function for reducing autocorrelations of the residuals from the predictions.
- Bayesian optimization of hyper parameters for faster prediction for ESN

THANK YOU