# Large-scale Probabilistic Forecasting in Energy Systems using Sparse Gaussian Conditional Random Fields

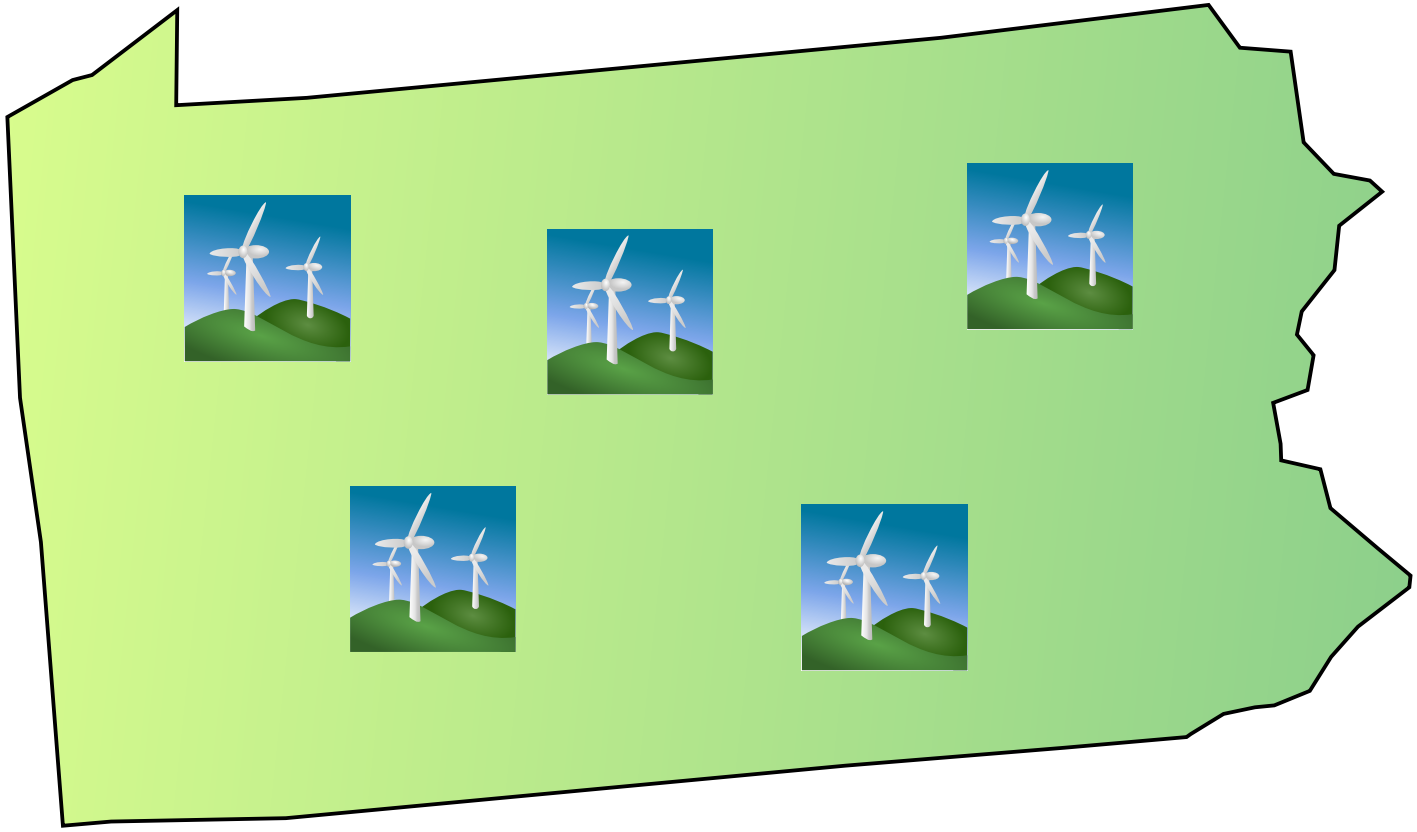
Matt Wytock, Zico Kolter
School of Computer Science
Carnegie Mellon University

December 10, 2013

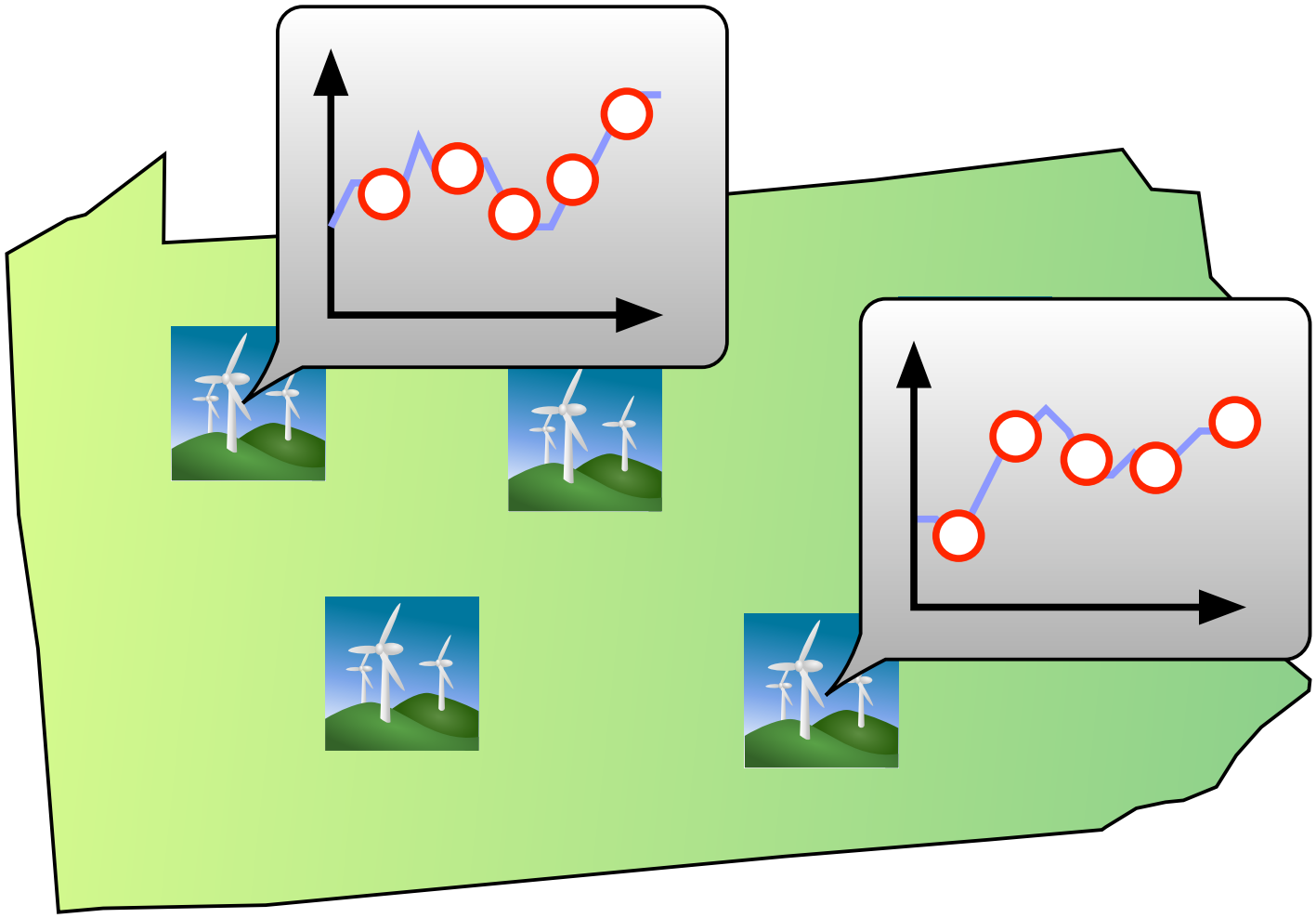
# Outline

1. Probabilistic forecasting problem
2. Fast algorithm for large-scale problems
3. Copula transform for non-Gaussian distributions
4. Wind power forecasting results

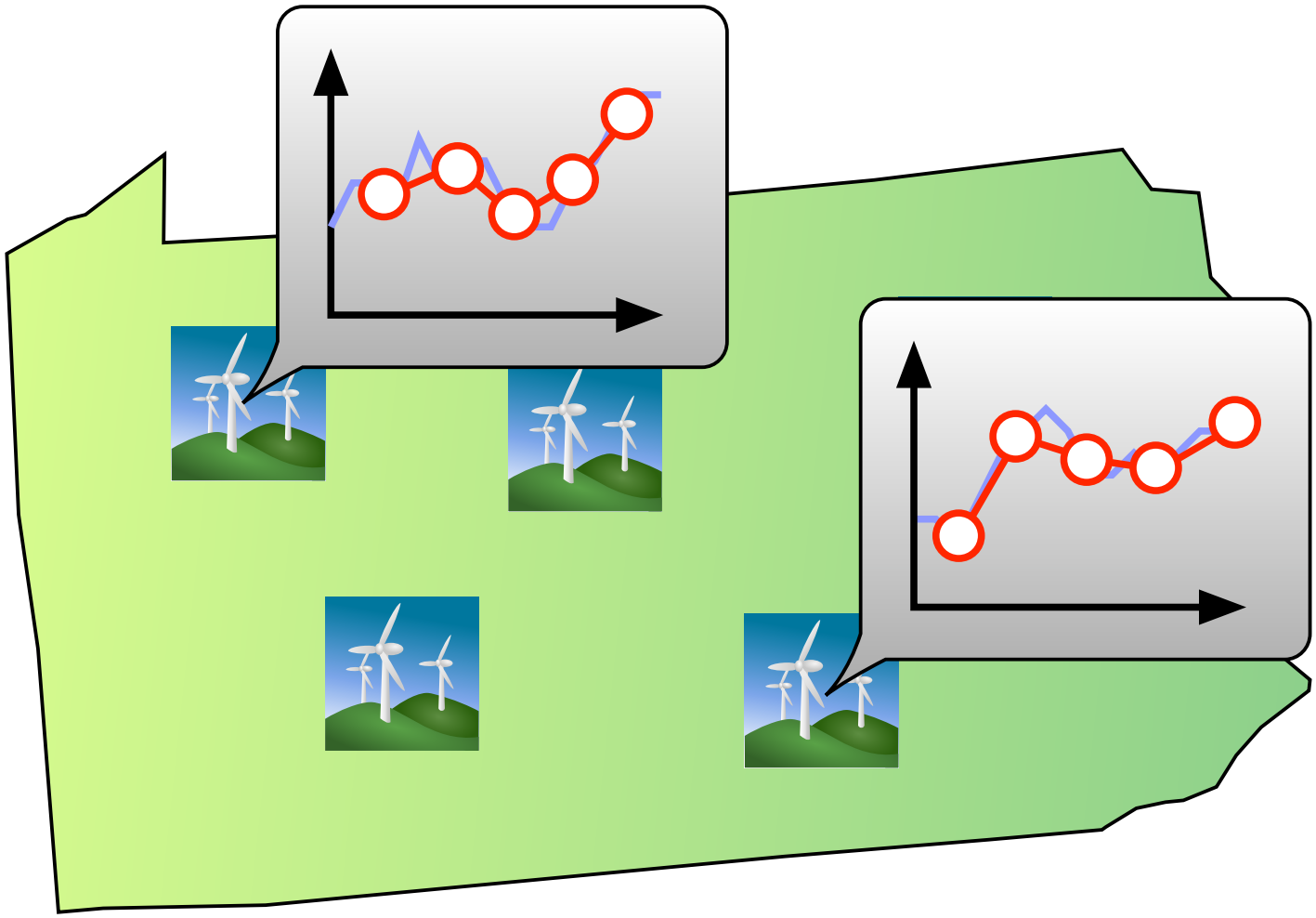
# Wind power forecasting



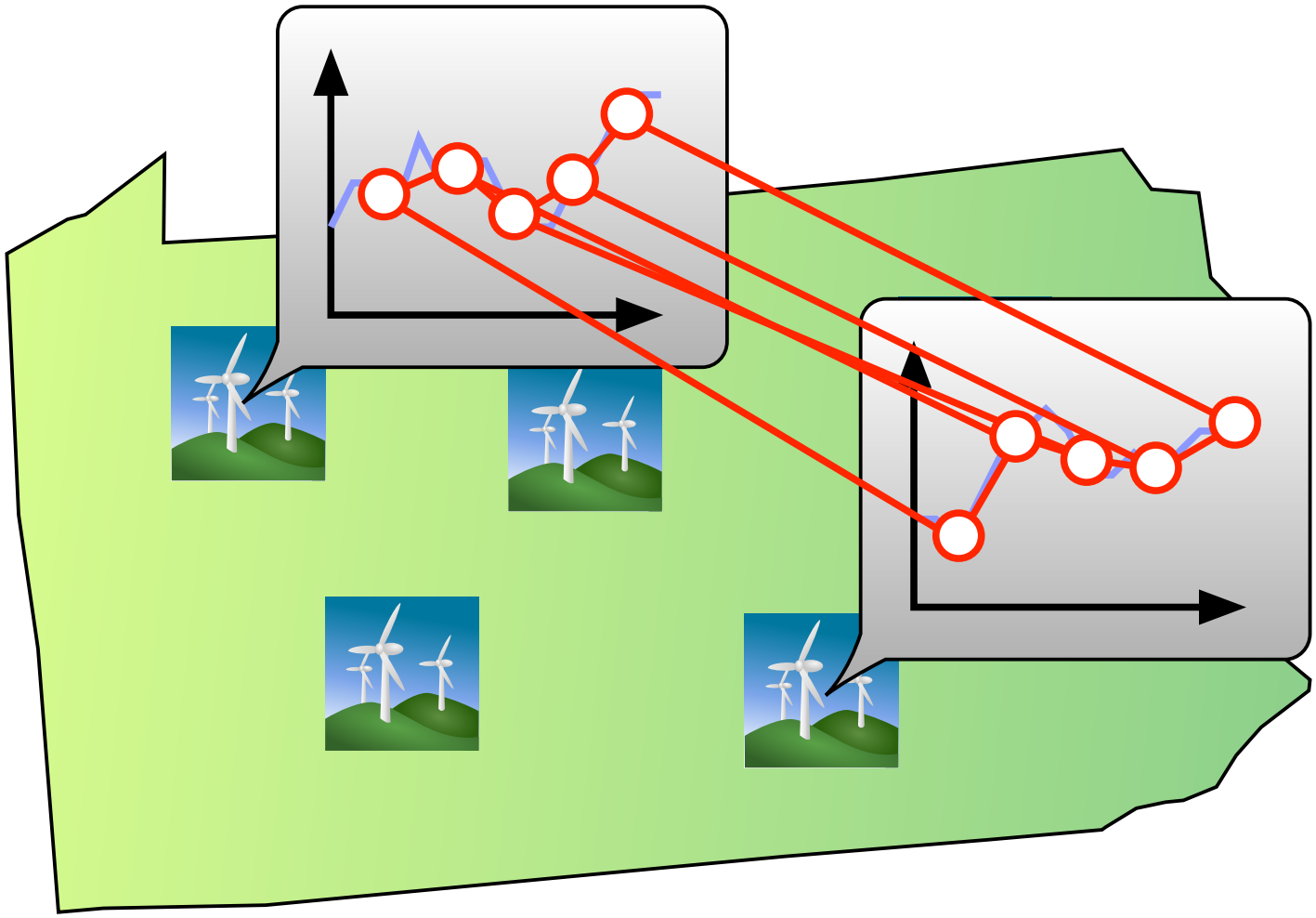
# Wind power forecasting



# Wind power forecasting

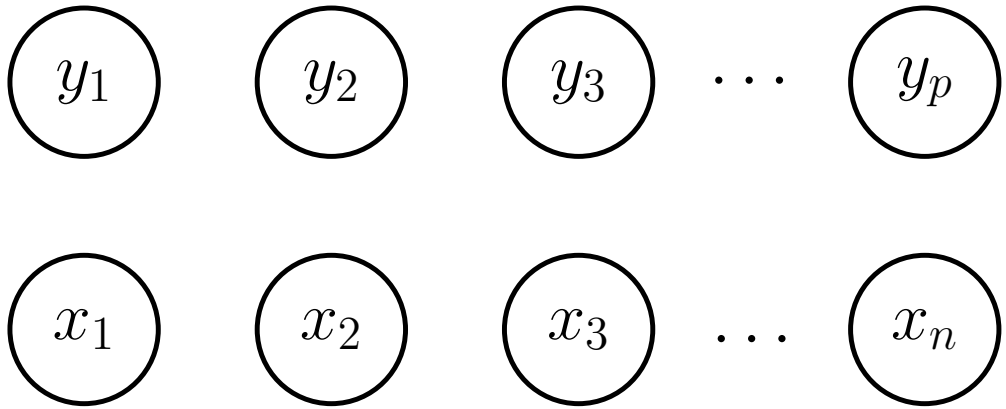


# Wind power forecasting



# The sparse Gaussian CRF model

Output: $y \in \mathbb{R}^p$, Input: $x \in \mathbb{R}^n$

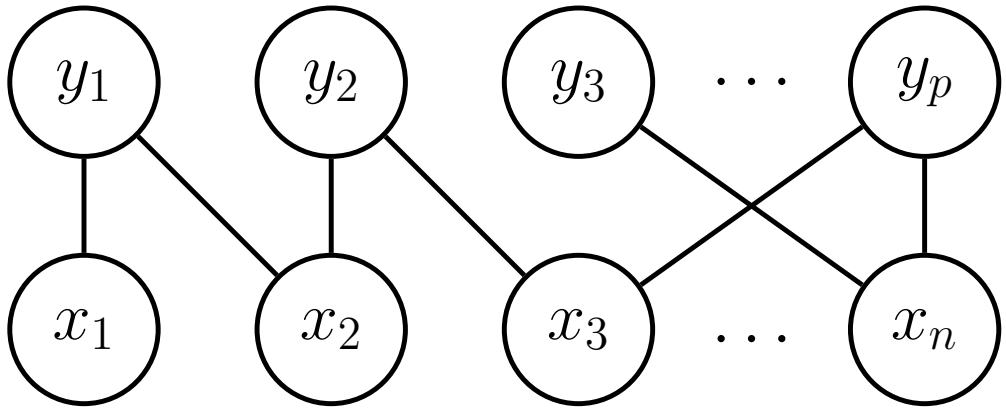



Example output: Future wind power

Example input: Past wind power, weather forecasts, etc.

# The sparse Gaussian CRF model

Output: $y \in \mathbb{R}^p$, Input: $x \in \mathbb{R}^n$

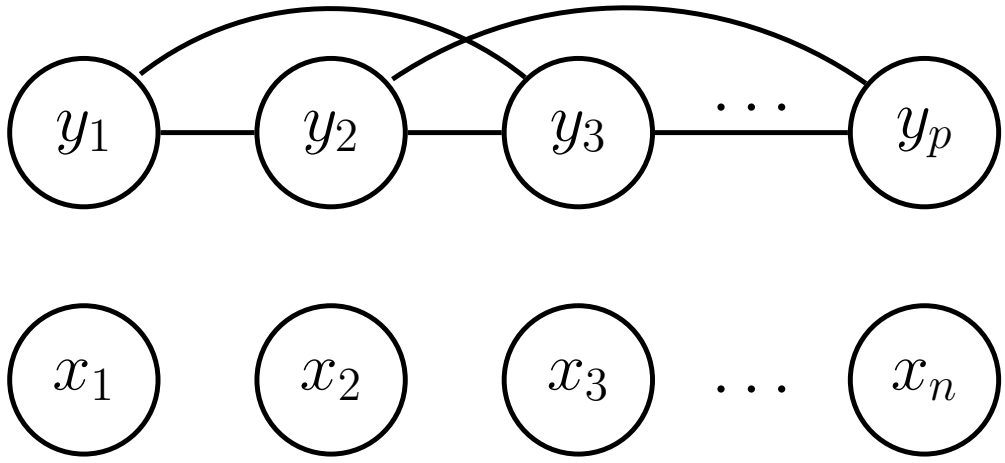



$$p(y|x) \propto \exp\left(-y^T y - 2x^T \Theta y\right)$$

$\Theta \in \mathbb{R}^{n \times p}$ models dependence of output on input, multivariate linear regression

# The sparse Gaussian CRF model

Output: $y \in \mathbb{R}^p$, Input: $x \in \mathbb{R}^n$

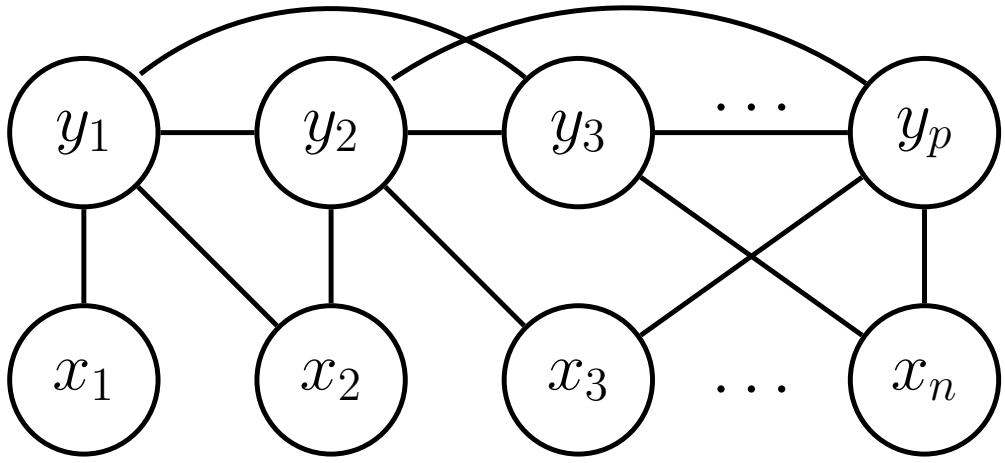



$$p(y|x) \propto \exp\left(-y^T \Lambda y\right)$$

$\Lambda \in \mathbb{R}^{p \times p}$ models dependencies in output, sparse inverse covariance estimation problem (e.g. Meinshausen and Bühlmann, 2006)

# The sparse Gaussian CRF model

Output: $y \in \mathbb{R}^p$, Input: $x \in \mathbb{R}^n$



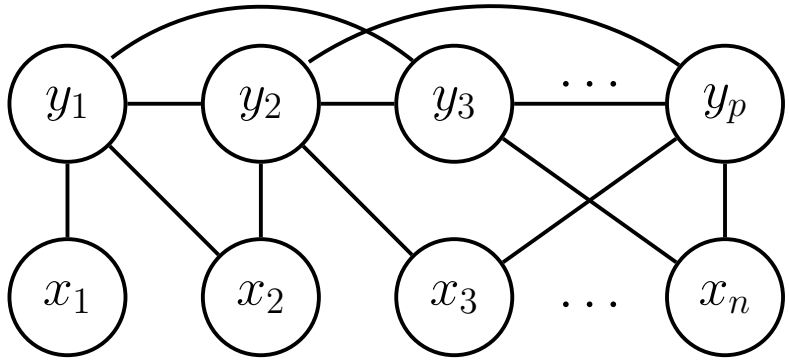

$$p(y|x) \propto \exp\left(-y^T \Lambda y - 2x^T \Theta y\right)$$

Conditional Gaussian distribution with parameterization that allows for *sparse* $\Lambda$ and $\Theta$ which is critical to scaling to large problems

# Outline

1. Probabilistic forecasting problem
2. Fast algorithm for large-scale problems
3. Copula transform for non-Gaussian distributions
4. Wind power forecasting results

# Optimization problem

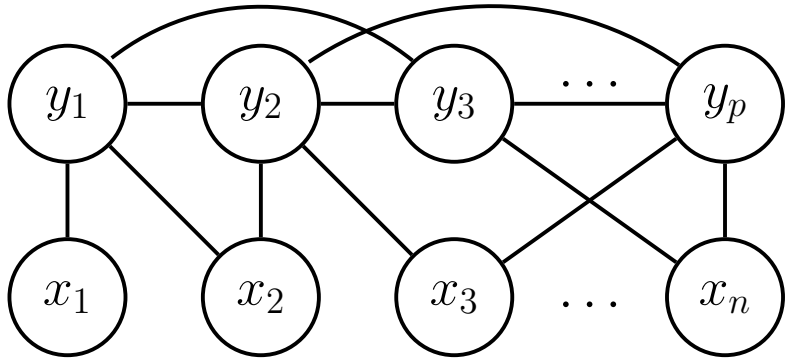



$$p(y|x) \propto \exp\left(-y^T \Lambda y - 2x^T \Theta y\right)$$

- Maximum likelihood estimation with $\ell_1$ regularization

$$\begin{aligned}\underset{\Lambda,\Theta}{\text{minimize}} & -\log|\Lambda| + \operatorname{tr}\Lambda S_{yy} + 2\operatorname{tr}\Theta S_{yx} + \operatorname{tr}\Lambda^{-1}\Theta^T S_{xx}\Theta \\ & + \lambda\|\Lambda\|_1 + \lambda\|\Theta\|_1\end{aligned}$$

# Optimization problem




$$p(y|x) \propto \exp\left(-y^T \Lambda y - 2x^T \Theta y\right)$$

- Maximum likelihood estimation with $\ell_1$ regularization

$$\underset{\Lambda,\Theta}{\text{minimize}} - \log|\Lambda| + \operatorname{tr}\Lambda S_{yy} + 2\operatorname{tr}\Theta S_{yx} + \operatorname{tr}\Lambda^{-1}\Theta^T S_{xx}\Theta$$
$$+ \lambda\|\Lambda\|_1 + \lambda\|\Theta\|_1$$

- Convex but difficult to optimize due to matrix fractional term

# Optimization: Newton coordinate descent

- State-of-the-art approach for sparse inverse covariance estimation, QUIC algorithm (Hsieh et al. 2011), see also Tseng and Yun, 2009

# Optimization: Newton coordinate descent

- State-of-the-art approach for sparse inverse covariance estimation, QUIC algorithm (Hsieh et al. 2011), see also Tseng and Yun, 2009

- Basic proximal Newton for minimizing $f(x) + \lambda\|x\|_1$, repeat
  1. Form the second-order Taylor expansion

$$\hat{f}(x + \Delta) = f(x) + \nabla_x f(x)^T \Delta + \frac{1}{2} \Delta^T \nabla_x^2 f(x) \Delta$$

  2. Solve for the regularized Newton step

$$d = \arg\min_{\Delta} \hat{f}(x + \Delta) + \lambda\|x + \Delta\|_1$$

  3. Update $x$ using backtracking line search

# Optimization: Newton coordinate descent

- State-of-the-art approach for sparse inverse covariance estimation, QUIC algorithm (Hsieh et al. 2011), see also Tseng and Yun, 2009

- Basic proximal Newton for minimizing $f(x) + \lambda\|x\|_1$, repeat
  1. Form the second-order Taylor expansion

$$\hat{f}(x + \Delta) = f(x) + \nabla_x f(x)^T \Delta + \frac{1}{2} \Delta^T \nabla_x^2 f(x) \Delta$$

  2. Solve for the regularized Newton step

$$d = \arg\min_{\Delta} \hat{f}(x + \Delta) + \lambda\|x + \Delta\|_1$$

  3. Update $x$ using backtracking line search

- Newton CD method solves step 2 using coordinate descent

# Newton CD details

- Actual second-order approximation for sparse Gaussian CRF

$$\hat{f}(\Lambda + \Delta_\Lambda, \Theta + \Delta_\Theta) = f(\Lambda, \Theta) + \operatorname{tr} S_{yy}\Delta_\Lambda + 2\operatorname{tr} S_{yx}\Delta_\Theta$$
$$- \operatorname{tr} \Lambda^{-1}\Delta_\Lambda + 2\operatorname{tr} \Lambda^{-1}\Theta^T S_{xx}\Delta_\Theta - \operatorname{tr} \Lambda^{-1}\Theta^T S_{xx}\Theta\Lambda^{-1}\Delta_\Lambda$$
$$+ \operatorname{tr} \Lambda^{-1}\Delta_\Lambda\Lambda^{-1}\Theta^T S_{xx}\Theta\Lambda^{-1}\Delta_\Lambda - \frac{1}{2}\operatorname{tr} \Lambda^{-1}\Delta_\Lambda\Lambda^{-1}\Delta_\Lambda$$
$$+ \operatorname{tr} \Lambda^{-1}{\Delta_\Theta}^T S_{xx}\Delta_\Theta - 2\operatorname{tr} \Lambda^{-1}\Delta_\Lambda\Lambda^{-1}\Theta^T S_{xx}\Delta_\Theta$$

# Newton CD details

- Actual second-order approximation for sparse Gaussian CRF

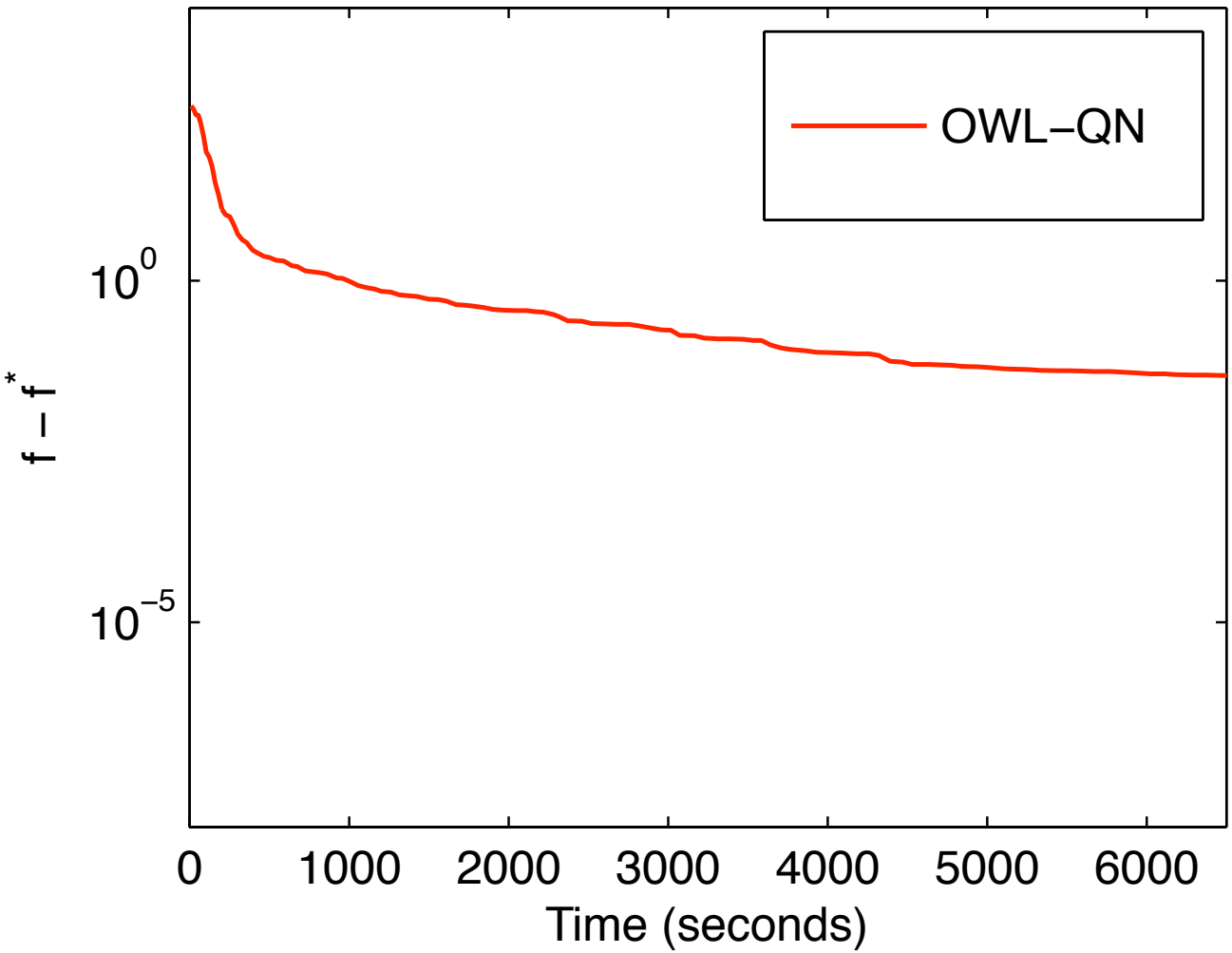
$$\begin{aligned}\hat{f}(\Lambda+\Delta_\Lambda,\Theta+\Delta_\Theta) &= f(\Lambda,\Theta) + \operatorname{tr} S_{yy}\Delta_\Lambda + 2\operatorname{tr} S_{yx}\Delta_\Theta \\ &- \operatorname{tr}\Lambda^{-1}\Delta_\Lambda + 2\operatorname{tr}\Lambda^{-1}\Theta^T S_{xx}\Delta_\Theta - \operatorname{tr}\Lambda^{-1}\Theta^T S_{xx}\Theta\Lambda^{-1}\Delta_\Lambda \\ &+ \operatorname{tr}\Lambda^{-1}\Delta_\Lambda\Lambda^{-1}\Theta^T S_{xx}\Theta\Lambda^{-1}\Delta_\Lambda - \frac{1}{2}\operatorname{tr}\Lambda^{-1}\Delta_\Lambda\Lambda^{-1}\Delta_\Lambda \\ &+ \operatorname{tr}\Lambda^{-1}\Delta_\Theta{}^T S_{xx}\Delta_\Theta - 2\operatorname{tr}\Lambda^{-1}\Delta_\Lambda\Lambda^{-1}\Theta^T S_{xx}\Delta_\Theta\end{aligned}$$

- Key to implementation is storing/updating products $\Delta_\Lambda\Lambda^{-1}$ and $\Delta_\Theta\Lambda^{-1}$ so that each coordinate descent step is $O(n+p)$

- Restrict minimization to *active set* of current nonzero coordinates plus those with gradient $> \lambda$

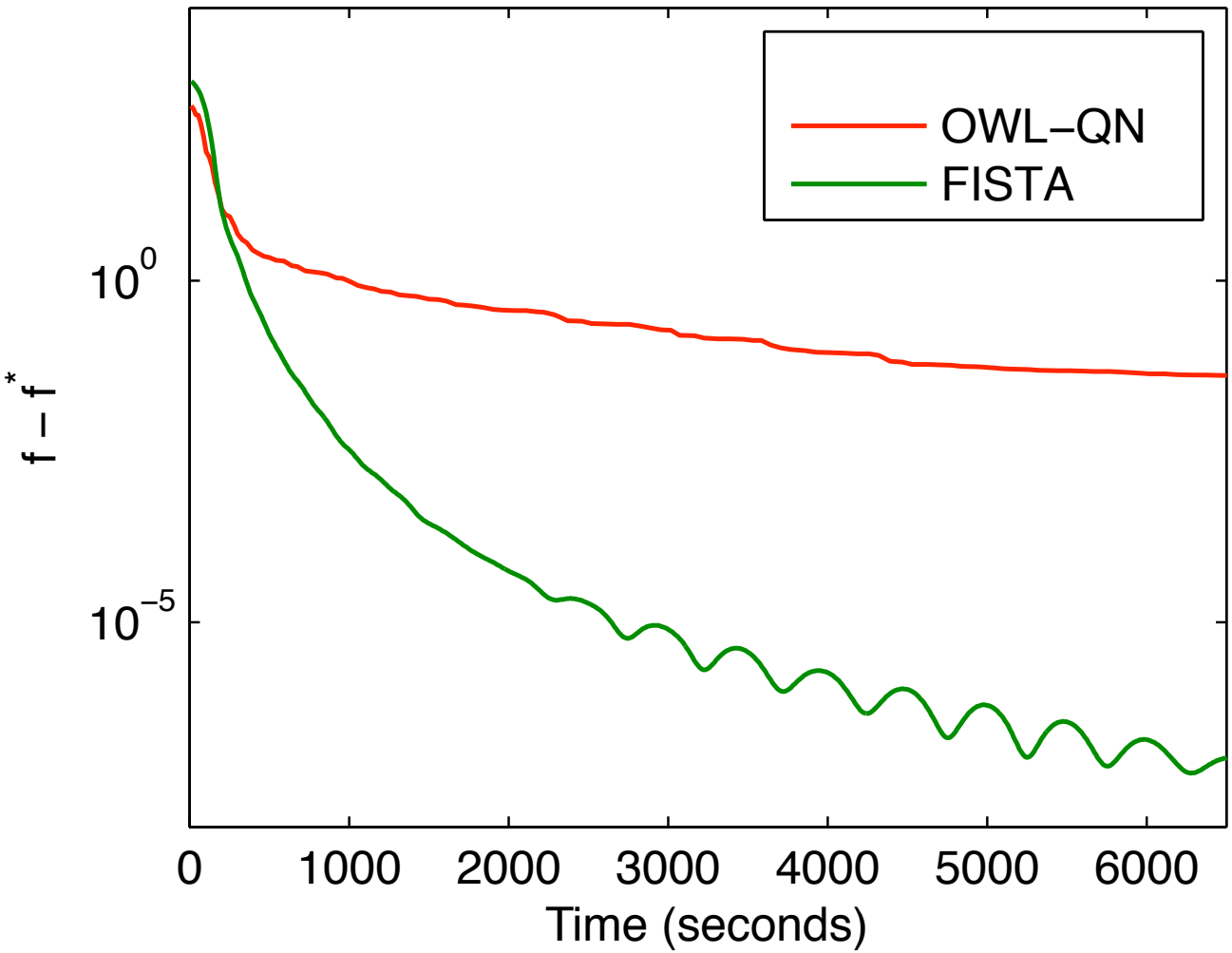
# Optimization performance

Synthetic data $n = 4000$, $p = 1000$



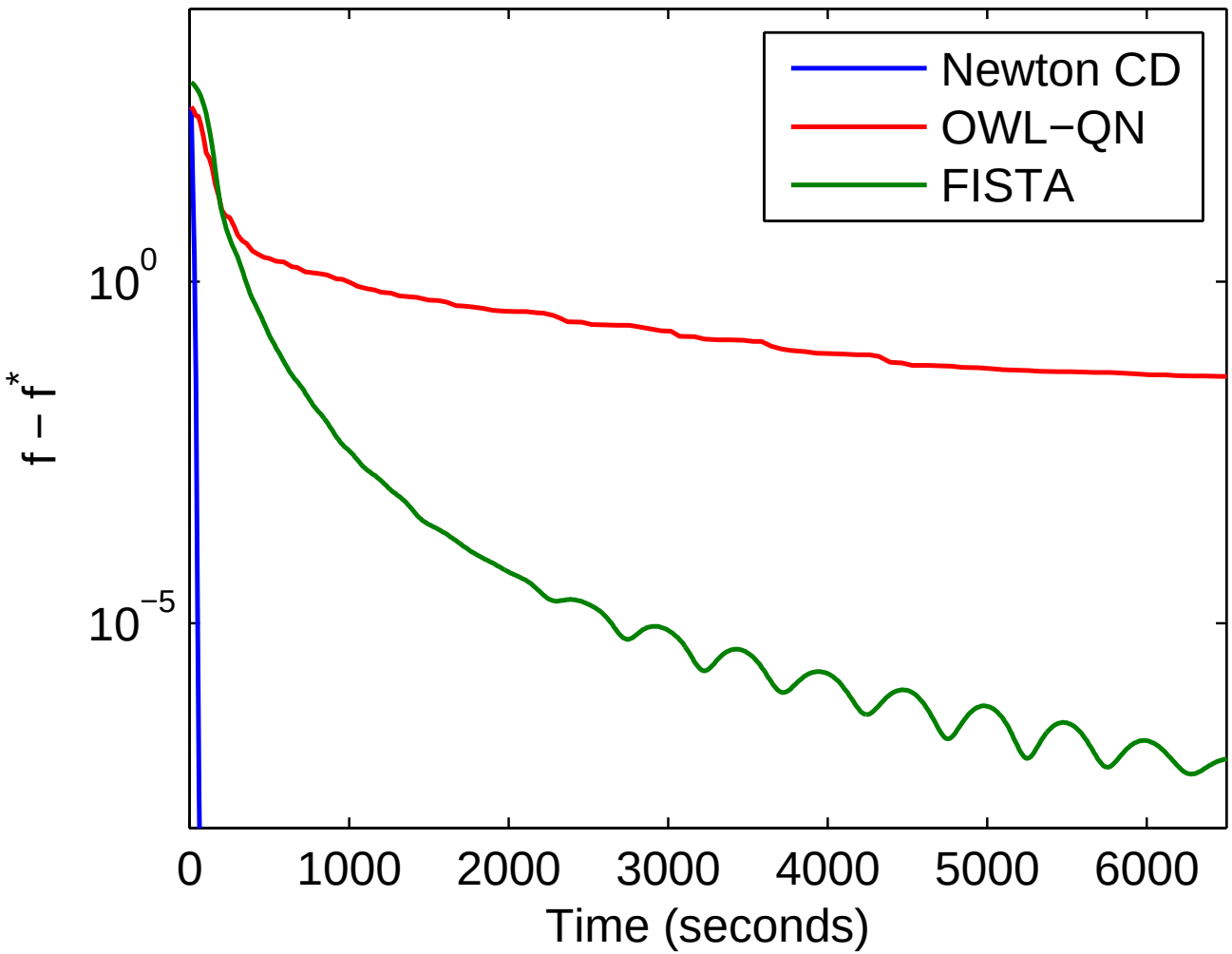
# Optimization performance

Synthetic data $n = 4000$, $p = 1000$



# Optimization performance

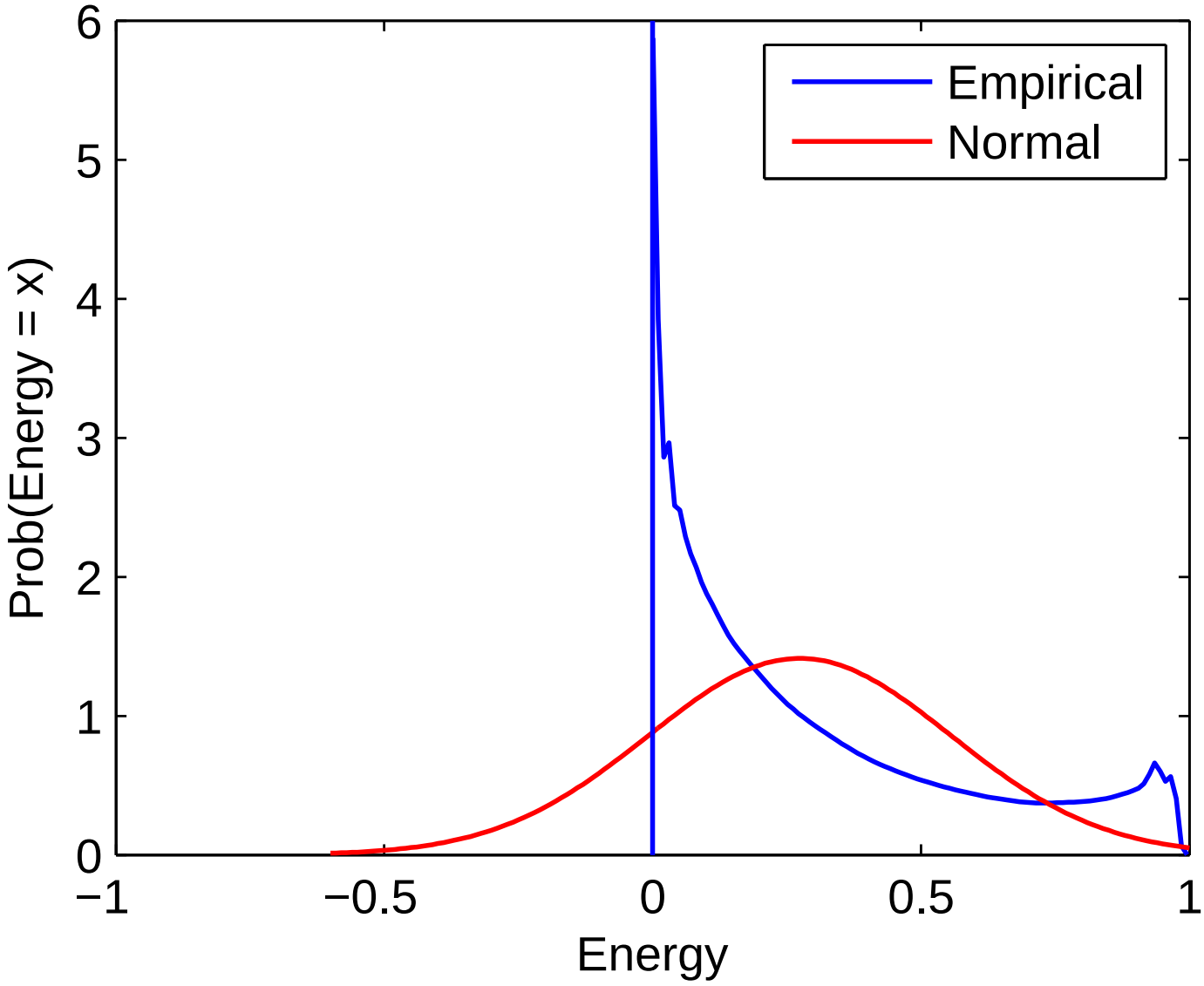
Synthetic data $n = 4000$, $p = 1000$



# Outline

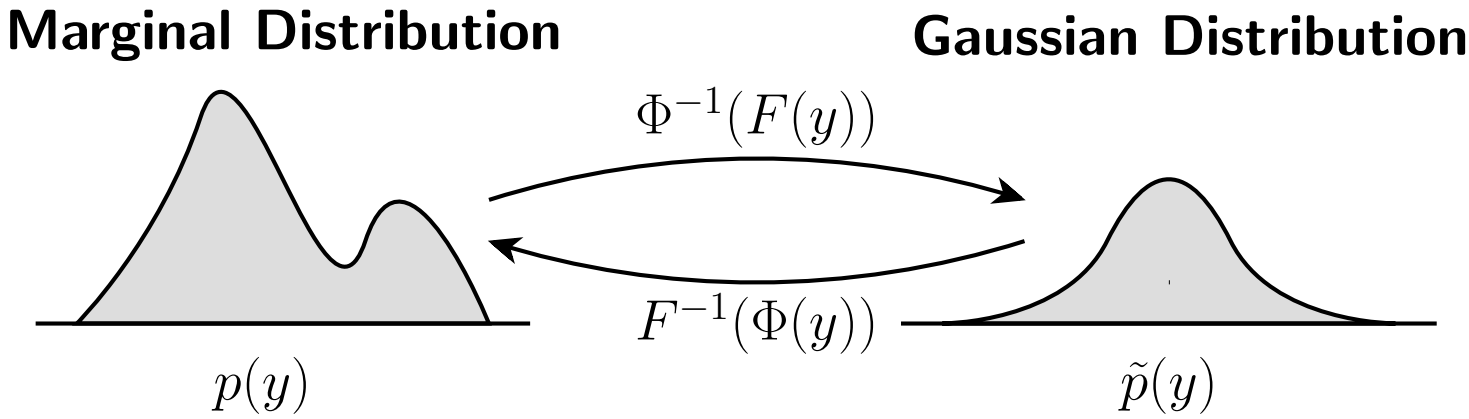
1. Probabilistic forecasting problem
2. Fast algorithm for large-scale problems
3. Copula transform for non-Gaussian distributions
4. Wind power forecasting results

# Empirical distribution of wind power



# Copulas for non-Gaussian distributions

- Simple way to model arbitrary univariate distributions




- For high-dimensional data we transform the marginals and model the dependence as multivariate Gaussian

# Fitting the model with copulas

To fit the model given training examples $(x_i, y_i)$, $i = 1, \ldots, m$

1. Apply the copula transform to the elements of each $y_i$

$$(\tilde{y_i})_j = \Phi^{-1}(F_j((y_i)_j))$$

2. Estimate $\Lambda$, $\Theta$ in the sparse Gaussian CRF with $(x_i, \tilde{y_i})$

# Predicting and sampling with copulas

Given a trained model, for a new input $x$ we can:

1. Compute the most likely scenario

$$\tilde{y} = -\Lambda^{-1}\Theta^T x$$

   or, sample from possible scenarios

$$\tilde{y} \sim \mathcal{N}(-\Lambda^{-1}\Theta^T x, \Lambda^{-1})$$

2. Apply inverse copula transform to the elements of $\tilde{y}$

$$\hat{y}_j = F^{-1}(\Phi(\tilde{y}_j))$$

# Outline

1. Probabilistic forecasting problem
2. Fast algorithm for large-scale problems
3. Copula transform for non-Gaussian distributions
4. Wind power forecasting results

# Wind power forecasting

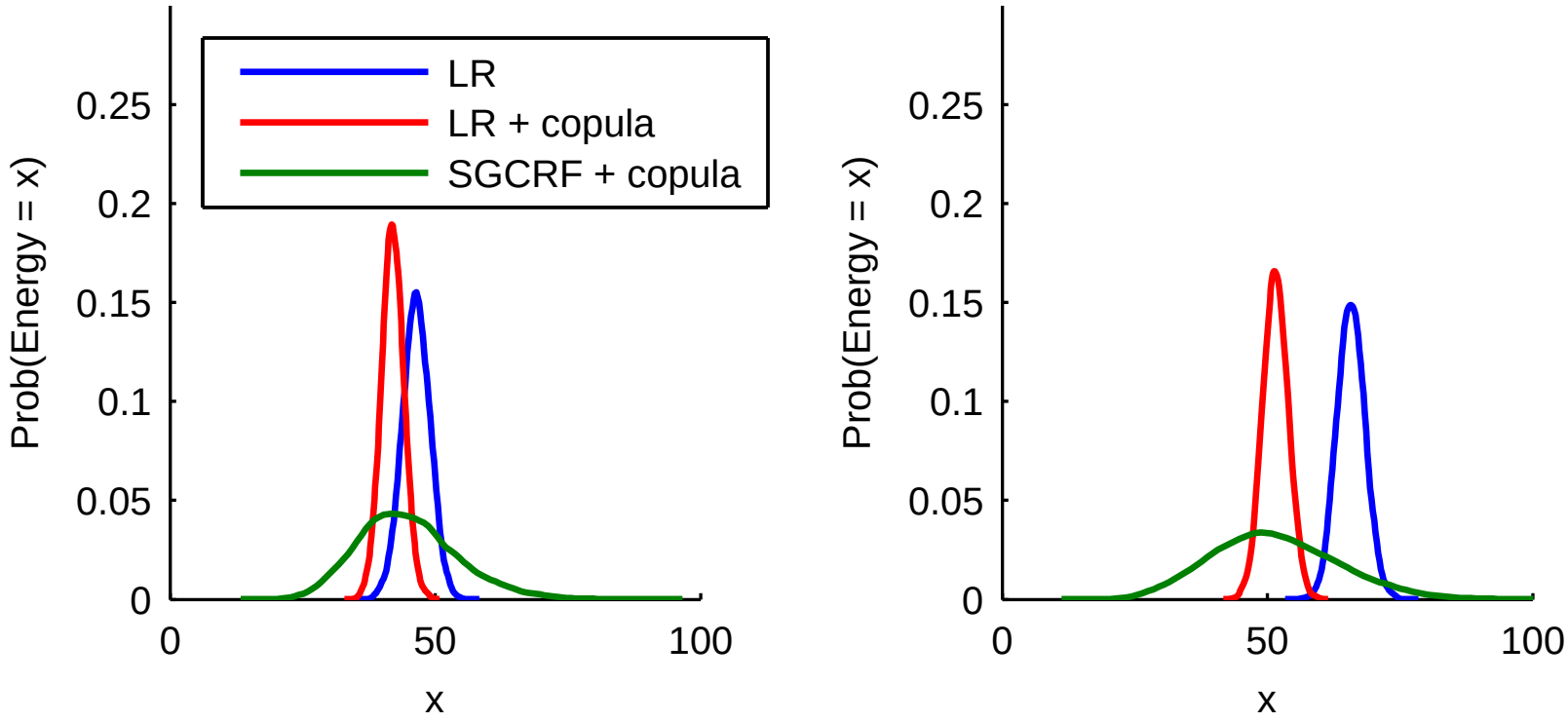
- Data from competition on Kaggle that ran in October 2012
- Outputs: wind power at 7 wind farms over 48 hours, $p = 336$
- Inputs: past 8 hours of wind power and 10 RBF features over wind forecasts, $n = 3417$
- Heavily optimized features for competition, resulting in a 5th place finish using ordinary least squares

# Comparison on mean-squared error

| **Algorithm** | **RMSE** |
|---|---|
| Least squares | 0.1560 |
| Least squares + copula | 0.1636 |
| ARMAX | 0.1714 |
| SGCRF | **0.1488** |
| SGCRF + copula | 0.1584 |

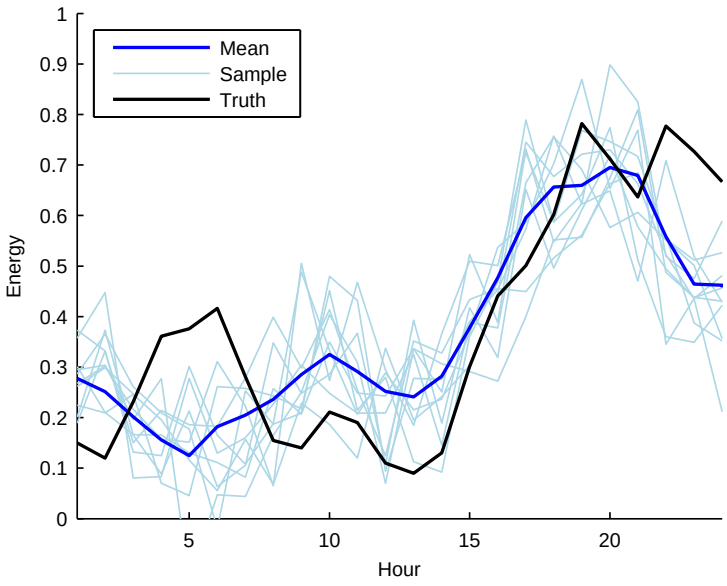
Significant improvement over least squares, larger than the difference between 5th and 1st place Kaggle finish

# Comparison of distributions

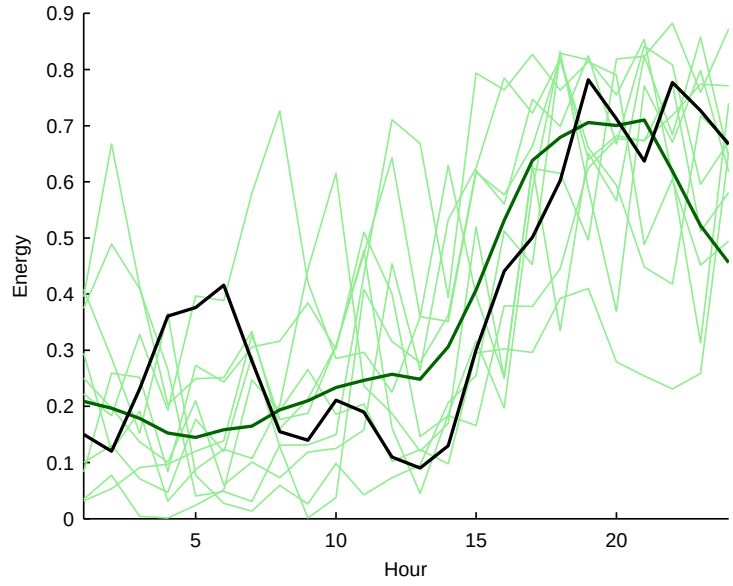



Real advantage comes from modeling the accurately modeling the *distribution* over future possible scenarios

# Comparison of samples




Least squares




Sparse Gaussian CRF

# Confidence interval coverage

We can use samples to construct confidence intervals and evaluate empirically how often this captures the true outcome

| **Algorithm** | **90%** | **95%** | **99%** |
|---|---|---|---|
| LS | 0.1944 | 0.2500 | 0.3333 |
| LS + copula | 0.2176 | 0.2639 | 0.3380 |
| ARMAX | 0.2454 | 0.3102 | 0.4213 |
| SGCRF | 0.4306 | 0.5370 | 0.6389 |
| SGCRF + copula | **0.8796** | **0.9259** | **0.9676** |

# Summary

- Sparse Gaussian CRFs provide an appealing framework for probabilistic forecasting
- Fast algorithm allows us to scale to large examples
- Copulas extend modeling to non-Gaussian distributions
- State-of-the art results in wind power forecasting