# Baseline Power Estimation

Ted Xiao
Hybrid Systems Laboratory
University of California, Berkeley
Email: tx@berkeley.edu

Sumedh Bhattacharya
Hybrid Systems Laboratory
University of California, Berkeley
Email: sumedh1234@berkeley.edu

## I. Introduction

Energy usage control for commercial buildings can reduce peak power consumption and stabilize overall power grid consumption by intelligently applying inputs to individual buildings. However, in order to apply intelligent control, an accurate tool is needed for calculating baseline power consumption. This value is difficult to predict to an hourly degree of accuracy without requiring building specific equipment. Specifically, in this paper we study HVAC (Heating, Ventilation, and Air Conditioning) systems in these commercial buildings as they are a major source of power consumption, and their power consumption patters follow a daily and seasonal trend. As commercial buildings account for more than 35% of the electricity consumption in the U.S. and 39% of this is due to HVAC systems, an application of this baselining technique would allow us to model 14% of the energy drawn on a power grid without using any additional hardware [1]. In this paper, we introduce three types of models: the first using only weather data and Lasso regression, the second utilizing an EM approach using time series forecasting that reduces error immensely, and finally a combination of both TSF on the power readings and the weather data using K-Nearest Neighbor and Recurrent Neural Networks to gain an even finer level of accuracy.

Initially, our first attempt tried to fit a correlation between publicly available weather data to understand the trends people may have in power usage and its correlation with different weather patterns. This was able to understand the general trend of the power series data but still had too high of an error to be used as a baseline value to apply control in future experiments on. The power readings we obtained from SDH are in a scale of 0-100 where the value represents the percentage of maximum power that is being used. For this, we wanted to be able to predict the power values to an accuracy of at least below 2%. We propose a generalization of thermodynamic linear models and time series forecasting to estimate the conditional dependence on prior power use observations specific to a building. We apply the Expectation Maximization algorithm to explore the conditional dependence between time steps without propagating errors that is caused by iterative prediction.

## II. Previous Work

Traditional thermodynamic models create a unique physical model for the building using metrics from different thermal and atmospheric sensors. These approaches leverage traditional regression techniques using equipment readings and measured power consumption. This allows for high accuracy but is heavily reliant on specialized equipment and cannot be applied throughout a city-wide power grid. Using the Sutardja Dai Hall building as an example of a commercial structure equipped with a variable air volume (VAV) HVAC system, existing models use measurements from the heating coils and pressure measurements from the air ducts to estimate the thermal baseline for the building.

$$\begin{aligned} x'(t) &= A_t(x(t)) + B_t q(x(t), u(t), v(t)) \\ y(t) &= Cx(t) \\ q(x(t), u(t), v(t)) &= q_{EW}(x(t), v(t)) + q_{win}(x(t), v(t)) \\ &\quad + q_{HVAC}(x(t), u(t), v(t)) + q_{IG}(t) \end{aligned} \tag{1}$$

where the state vector $x$ is the predicted temperatures of different regions in the building. $q_{EW}$ represents convective heat transfer and solar gains to the exterior walls, $q_{win}$ represents convective and solar radiation gains through the windows, $q_{HVAC}$ is a negative heat gain due to the HVAC system and $q_{IG}$ represents internal gains due to occupancy, equipment and other unmodeled uncertainties. $y$ is then a vector containing the averaged results for the predicted temperatures in different groups of room. This allows us to build a highly accurate thermal model and use as the baseline for the thermal temperatures in the room while applying control, and then we can calculate the deviations from the expected baseline values.

This control problem is highly reliant on installing the necessary sensors and cannot be generalized to applying predictive control across a power grid when only 30% of the commercial buildings have the variable air volume (VAV) HVAC system. It also allows us to calculate not just the deviation due to control from the expected temperature comfort zones in a building but also the difference in consumption caused by it. Therefore, we use a similar model to baseline power consumption which we can calculate to the expected power consumption for the next day (24 hours) to a high degree of accuracy without requiring specialized equipment by instead choosing to use regional weather data. We augment our loss of accuracy by using region-specific instead of building-specific data, by understanding the evolution of the building's historical power usage using time series forecasting.

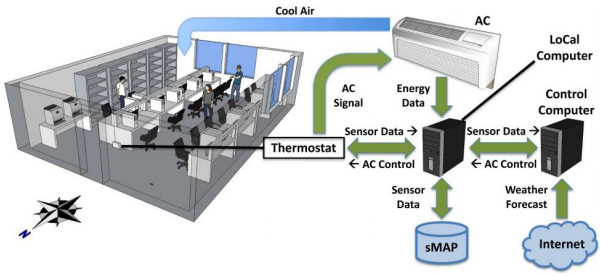
Support vector machines (SVMs) have been widely used in the past. Dong et al. [4] made an early attempt in using SVMs in building energy consumption prediction. Their model used weather data and monthly utility bills as input to predict annual building energy consumption in tropical regions, and reported a percentage error within 4%. An approach using multiagent systems at USC [2] reported a percentage error of 2.8%. We aim to reach these levels of accuracy for our prediction of the

HVAC power consumption by the main supply fans in SDH but our model does not currently support the entire HVAC system and has not been tested on multiple buildings.

## III. Data

### A. BRITE Data Preprocessing

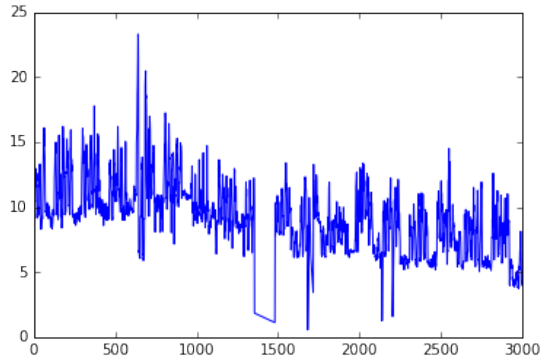
Our data consists of historic data collected from the Berkeley Retrofitted and Inexpensive HVAC Testbed for Energy Efficiency (BRITE) at Sutardja Dai Hall.




BACnet is a communication protocol that is used for building automation and control networks. Specifically, it allows for the control of HVAC systems, lights, and other electronics to allow building automation to communicate information between one another regardless of the specific service the building performs.

Using BACnet, we were able to retrieve the building's power consumption in its two main supply fans, known as AHUs (air handling units) for a period of five months, from 2014-03-15 to 2015-08-15 in minute intervals. We then average these values by hour to match our forecast data. These readings were then preprocessed to have zero mean and unit variance before being entered into our time series models. To adjust for times where the fans were manually shut off (which is a rare event that should not happen during normal functioning), we see negative readings. To compensate for this, we use linear regression to replace the values between two different valid, positive readings. These values are all then stored in a dictionary keyed by time.

An example view of our data is show below:




### B. Forecast Data Preprocessing

Our weather forecast data is retrieved from forecast.io's Dark Sky API which offers both hourly and daily values for a given latitude and longitude. From their hourly readings, we obtain the following features:

- precipitation intensity, probability and type
- temperature
- apparent temperature
- humidity
- wind speed
- cloud cover

We also add to this the number of the week to account for seasonality and a binary indicator of if it is a weekend or not. Each feature in this $(3000, 10)$ array, with 3000 hourly readings (hours with missing features are dropped), is then preprocessed to have $0$ mean and unit variance.

All of the data provided by forecast.io does not have every available feature for the BRITE Dataset due to technical issues during that time or instrument malfunction. To account for this, all values are stored in a dictionary keyed by time. If there are any times that are missing, we also skip the appropriate key for that time in our time series data. However, this would normally cause a discontinuity within the temporal relation of a day's power readings. To ensure consistency across all of the days we are predicting on, we choose to skip that day's values entirely to ensure that we only learn on days that we have full weather data for.

### C. Loss Metric Selection

For picking out our metric to measure the accuracy of our model, we looked at Zhang's study [2] that found that proposed metrics for solar power forecasting fell largely into statistical metrics for different temporal and geographical scales, uncertainty quantification metrics, ramp characterization metrics and economic metrics. From the first category, mean absolute error was found suitable for evaluating uniform forecast error across a span of a day and this is the metric we used to measure the accuracy of our models.

## IV. Models

From the temporal point of view, the simplest approach to estimate forecasting baselines is that of climatology. The climatology approach consists of using a constant long-term average value throughout the entire forecasting period. This average value is then used as a benchmark for the forecasting skill with minimal effort. However, since building-specific power forecasting has surpassed this greatly, we instead approximate the baseline trend using only our weather matching model. We see that the weather model is able to approximate the general seasonal trends, and we attempt to improve on that model using the EM approach and the KNN and RNN approaches.

## V. Weather Matching Approach

The idea behind the weather matching apporach is to essentially create a matrix $A$, in which each row $a_p$ is a difference measurement for each of our weather features between the forecasted values for the day we wish to predict and the historical day, $p$, we have observed. Each of these differences then has a weight stored in $x$ it used to predict the absolute mean corresponding difference in power values between day $p$

and the future day $f$ we wish to predict. While training, $f$ is simply another historical day and we calculate the correlation between the change in weather data and change in power readings. So with n days, this gives us $n * (n-2)$ points to train on.

Specifically, the Lasso Model is structured with features $w_p = [w_{p1} \quad w_{p2} \quad ... \quad w_{p10}]$, where $w_{p1}$ is precipitation intensity, $w_{p2}$ is precipitation probability, and so on according to Section III.B. Similarly, for another day $f$, we have $w_f = [w_{f1} \quad w_{f2} \quad ... \quad f_{10}]$.

Defining $a_p$ and $\gamma_p$, we find:

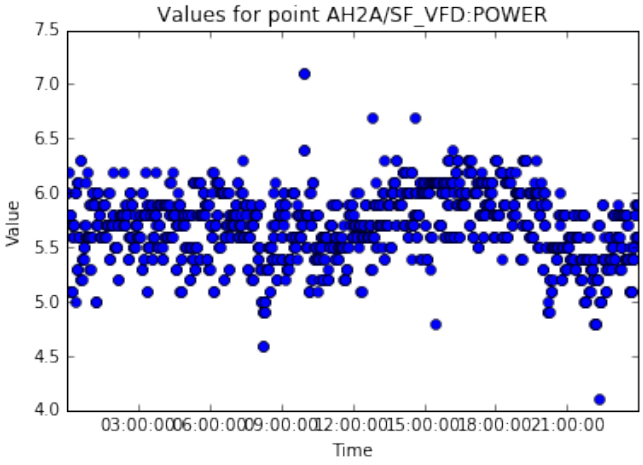
$$\begin{aligned} a_p &= w_f - w_p \\ a_{p+1} &= w_f - w_{p+1} \\ a_j &= w_f - w_j \forall j \in (p, p+n) \\ q &= n * (n-1)/2 \end{aligned} \tag{2}$$

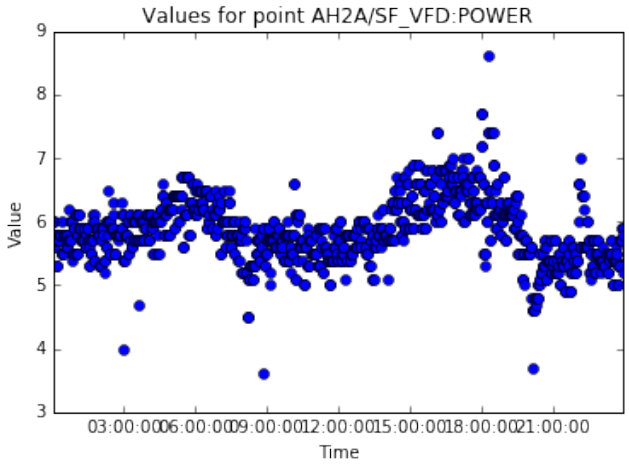
$$\begin{bmatrix} - & - & a_p & - & - \\ - & - & a_{p+1} & - & - \\ & & \vdots & & \\ - & - & a_{p+q} & - & - \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_1 0 \end{bmatrix} = \begin{bmatrix} y_p \\ y_{p+1} \\ \vdots \\ y_{p+q} \end{bmatrix}$$

$$\begin{aligned} \gamma_p &= [\gamma_{p,1} \quad \gamma_{p,2} \quad ... \quad \gamma_{p,24}] \\ \gamma_f &= [\gamma_{f,1} \quad \gamma_{f,2} \quad ... \quad \gamma_{f,24}] \\ y_p &= \text{mean}(\text{abs}(\gamma_p - \gamma_f)) \end{aligned}$$

Where $\gamma_p$ are historical power readings for day $p$ and $\gamma_f$ are the power readings for day $f$. After training to retrieve our $x_1 ... x_{10}$ weights, we pass in the weather differences between $w_{f'}$, our future forecasted date.

For example, over the course of a 30 days, from 2015-08-10 to 2015-09-10, it found the following two days to be the most similar:




When we want to make a prediction, $w_{f'}$ stores the forecasted days weather values and we create a new matrix $A_n$ using differences between $w_{f'}$ and all of our historical data. Computing $y_n$ using our $x$ weights correspondingly will now be the predicted mean absolute differences in power.

We then train this using linear regression with Lasso Regularization and obtain the following $x$ coefficients for differing number of components in PCA dimension reduction as well as their explained variance ratio:

- Predicted Coefficients for # of Components = 2
  [-0.60675826 -1.1959114 ]
  Explained Variance Ratio:
  [ 0.34663014 0.23113965]
  Accuracy: 17.62
- Predicted Coefficients for # of Components = 3
  [-0.60675826 -1.1959114 -0.60711793]
  Explained Variance Ratio:
  [ 0.34663014 0.23113965 0.14911575]
  Accuracy: 18.70
- Predicted Coefficients for # of Components = 4
  [-0.60675826 -1.1959114 -0.60711793 0.50238137]
  Explained Variance Ratio:
  [ 0.34663014 0.23113965 0.14911575 0.08784113]
  Accuracy: 18.73
- Predicted Coefficients for # of Components = 5
  [-0.60675826 -1.1959114 -0.60711793 0.50238137 0.84119556]
  Explained Variance Ratio:
  [ 0.34663014 0.23113965 0.14911575 0.08784113 0.06818651]
  Accuracy: 16.43
- Predicted Coefficients for # of Components = 6
  [-0.60675826 -1.1959114 -0.60711793 0.50238137 0.84119556 -1.2150931 ]
  Explained Variance Ratio:
  [ 0.34663014 0.23113965 0.14911575 0.08784113 0.06818651 0.03672205]
  Accuracy: 15.55
- Predicted Coefficients for # of Components = 7
  [-0.60675826 -1.1959114 -0.60711793 0.50238137 0.84119556 -1.2150931 -1.31080333]

Explained Variance Ratio:
[ 0.34663014 0.23113965 0.14911575 0.08784113 0.06818651 0.03672205 0.03029435]
Accuracy: 14.15

- Predicted Coefficients for # of Components = 8
[-0.60675826 -1.1959114 -0.60711793 0.50238137 0.84119556 -1.2150931 -1.31080333 -0.1864261 ]
Explained Variance Ratio:
[ 0.34663014 0.23113965 0.14911575 0.08784113 0.06818651 0.03672205 0.03029435 0.02658181]
Accuracy: 14.74

- Predicted Coefficients for # of Components = 9
[-0.60675826 -1.1959114 -0.60711793 0.50238137 0.84119556 -1.2150931 -1.31080333 -0.1864261 -0.96710048]
Explained Variance Ratio:
[ 0.34663014 0.23113965 0.14911575 0.08784113 0.06818651 0.03672205 0.03029435 0.02658181 0.0225175 ]
Accuracy: 14.15

- Predicted Coefficients for # of Components = 10
[-0.60675826 -1.1959114 -0.60711793 0.50238137 0.84119556 -1.2150931 -1.31080333 -0.1864261 -0.96710048 -7.26094998]
Explained Variance Ratio:
[ 0.34663014 0.23113965 0.14911575 0.08784113 0.06818651 0.03672205 0.03029435 0.02658181 0.0225175 0.00082787]
Accuracy: 12.07

As we can see, dimensionality reduction does result in a loss in data but due to the increasing $n^2$ rate of the training size based on $n$ days, it is helpful to note that we can reduce our dimensionality to 7 components with only a 2% cost to accuracy. Our full ten dimensional model, however, does guarantee us an accuracy of 12.07%, which is the mean absolute error for each reading in proportion to the expected reading. This is still much above industry standards and we aim to lower it much further. However, this is evidence that weather data alone can model general seasonal trends in power readings, even at the hour level granularity.

## VI. Expectation Maximization Approach

Now, we switch to an expectation maximization approach to use time series forecasting to understand the conditional dependence between a historical 24-hour measurement (the recorded power readings for the day before the one we wish to predict, $d-1$) and the following 24-hours (the predicted power readings of the day $d$). We project the time series data points to a high dimensional regressor space where their density can then be modelled using Gaussian Mixture Models (GMMs). This estimate of the probability density then allows us to perform short-to-medium term prediction by finding the conditional dependencies of the unknown values.

### A. Training

Taking a time series $z$ consisting of $n$ hourly power readings:

$$z_0, z_1, z_2, ..., z_{n-2}, z_{n-1}$$

We then conduct a delay embedding to create slices of window size $d$ and arrange them in a design matrix $X$:

$$X = \begin{bmatrix} z_0 & z_1 & \dots & z_{d-1} \\ z_1 & z_2 & \dots & z_d \\ \vdots & \vdots & & \vdots \\ z_{n-d} & z_{n-d+1} & \dots & z_{n-1} \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-d} \end{bmatrix}$$

The density of these points can then be modelled as a Gaussian Mixture Model with the following probability density function:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

where $\mathcal{N}(x|\mu_k, \Sigma_k)$ is the probability density function of the multivariate normal distribution, $\mu_k$ represents the means, $\Sigma_k$ the covariance matrices, and $\pi_k$ the mixing coefficients for each component $k (0 < \pi_k < 1, \sum_{k=1}^{K} \pi_k = 1)$.

Here the number of components, which is the number of Gaussian Mixtures, can be seen as the number of different types of days we are learning. The means of each Gaussian Mixture represents the average values for the power reading over the course of that type of day. The conditional probabilities then can be intuitively viewed as a measure of how much the given time slice fitted one of the historical days we have seen. The larger number of components, the greater the different types of days we believe existed in the past.

We train this algorithm using the EM algorithm for finding a maximum-likelihood fit given by

$$\log \mathcal{L}(\theta) = \log p(X|\theta) = \sum_{i=1}^{N} \log(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k))$$

where $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ are the necessary statistics that define the model.

The E-step is then to find the expected value for the log likelihood with latent variables given our design matrix and the estimation of $\theta^{(t)}$, which is the estimate of $\theta$ at timestep $t$ given by:

$$Q(\theta|\theta^{(t)}) = E_{Z|X,\theta^{(t)}}[\log \mathcal{L}(\theta; X, Z)]$$

This then requires us to calculate the probability, $r_{ik}$, that sample $x_i$ is generated by the $k$th Gaussian mixture before we can maximize the log likelihood in the M-step.

$$\begin{aligned} r_{ik}^{(t)} &= \frac{\pi_k^{(t)} \mathcal{N}(x_i|\mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_{j=1}^{K} \pi_{j=1}^{(t)} \mathcal{N}(x_i|\mu_j^{(t)}, \Sigma_j^{(t)})} \\ \theta^{(t+1)} &= \arg\max_{\theta} Q(\theta|\theta^{(t)}) \end{aligned} \tag{4}$$

We can now estimate our parameters for timestep $t+1$ with $N_k$ being the number of samples in the $k$th component:

$$\begin{aligned}\mu_k^{(t+1)} &= \frac{1}{N_k}\sum_{i=1}^{N} t_{ik}^{(t)} x_i \\ \Sigma_k^{(t+1)} &= \frac{1}{N_k}\sum_{i=1}^{N} t_{ik}^{(t)} - \mu_k^{(t+1)})(x_i - \mu_k^{(t+1)})^T \\ \pi_k^{(t+1)} &= \frac{1}{N_k}\sum_{i=1}^{N} t_{ik}^{(t)}\end{aligned} \tag{5}$$

The E and M step are then alternately repeated until convergence.

### B. Predicting

Now, to predict the next future 24 hours $F$ (unknown) we use the past 24 hour values $P$ (known). First, we split the means and covariances of each component into it's past and future components

$$\mu_k = \begin{bmatrix}\mu_k^P \\ \mu_k^F\end{bmatrix}, \Sigma_k = \begin{bmatrix}\Sigma_k^{PP} & \Sigma_k^{PF} \\ \Sigma_k^{FP} & \Sigma_k^{FF}\end{bmatrix}$$

We can find the probability that a sample $x_i^P$ with only past values known belongs to a component $k$

$$t_{ik} = \frac{\pi_k \mathcal{N}(x_i^P | \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i^P | \mu_j, \Sigma_j)}$$

Using this, we can find the conditional expectation of the future values originating from component $k$'s $F$ values and make a prediction $\hat{y}_i$:

$$\begin{aligned}\tilde{y}_{ik} &= \mu_k^F + \Sigma_k^{FP}(\Sigma_k^{PP})^{-1}(x_i^P - \mu_k^P) \\ \hat{y}_i &= \sum_{k=1}^{K} t_{ik}\tilde{y}_{ik}\end{aligned} \tag{6}$$

### C. Selecting Number of Components

To calculate the number of components, we had a choice to either pick the value that provides the lowest Akaike information criterion (AIC), given by

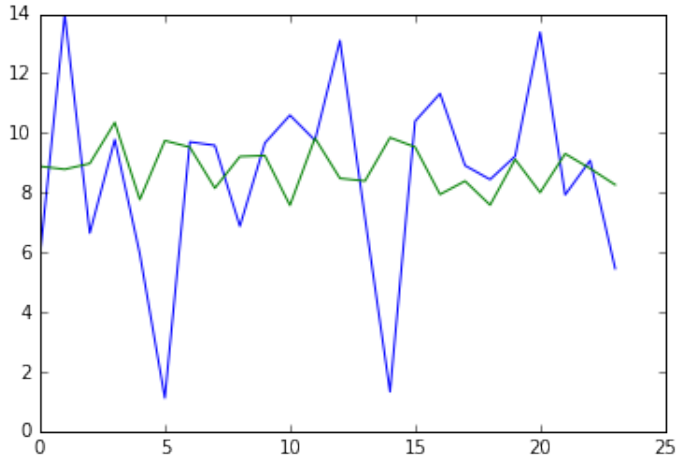
$$\text{AIC} = -2\log\mathcal{L}(\theta) + 2P$$

where $P = KD + \frac{1}{2}Kd(d+1) + K - 1$

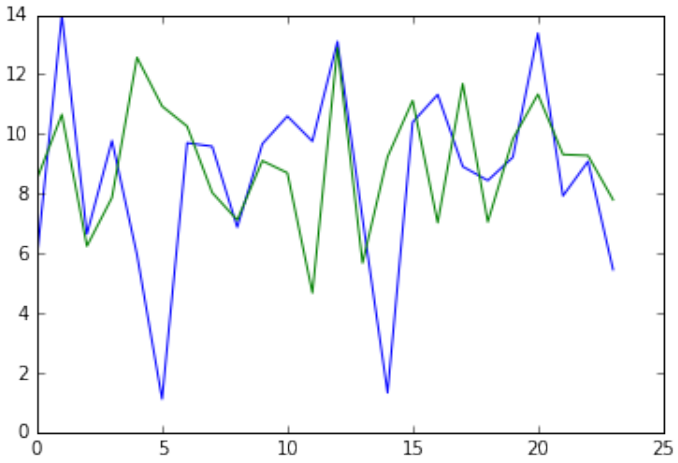
or to find the number of components which gives the least absolute mean error on the 20% of points held out for validation. We chose the latter for our model selection using 2360 slices of size 48 for training our model and 591 slices of size 24 for our validation set.

Intuitively, increasing the number of components increases the variation in our prediction which must be balanced so as not to overfit the training data. We received our lowest error with 40 components at 2.4041669.

To visually show the difference in variation, here we see an example of a fitting with 10 components:




and here is an example of a fitting with 40 components:




We also wanted to understand what would happen if all of our concentrations were not weighted equally. For this, we used a Gaussian Mixture with a Dirichlet process prior fit with variational inference to add regularization in order to integrate information from prior distributions. Specifically, a low value of the concentration prior prioritizes the weight of a few components while the rest are close to zero, while a high concentration prior has more highly active components. Here, the number of specified components becomes an upper bound while the actually active components in the model are much fewer. Intuitively, this allows us to pick a set of mean days that are the most representative of usual weekly behavior and will be the most active concentrations. This has both advantages and disadvantages.

Advantages:

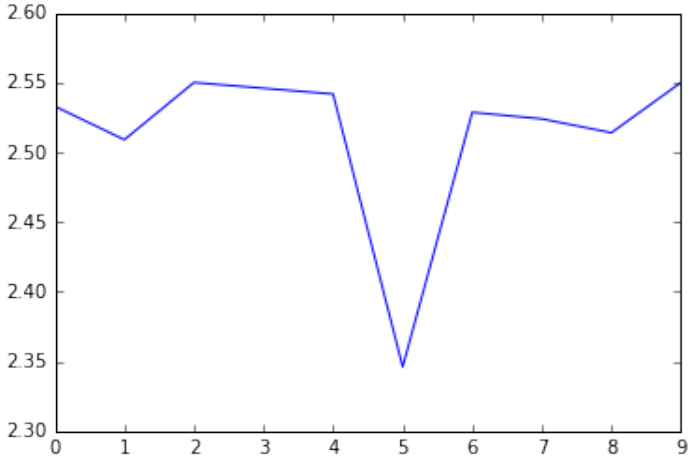
- Automatic component selection as a small weight concentration prior will only select a subset of the components to be active

- Less variation caused by differing number of components
- Less prone to overfitting as outlier values are regularized by lower weights

Disadvantages:

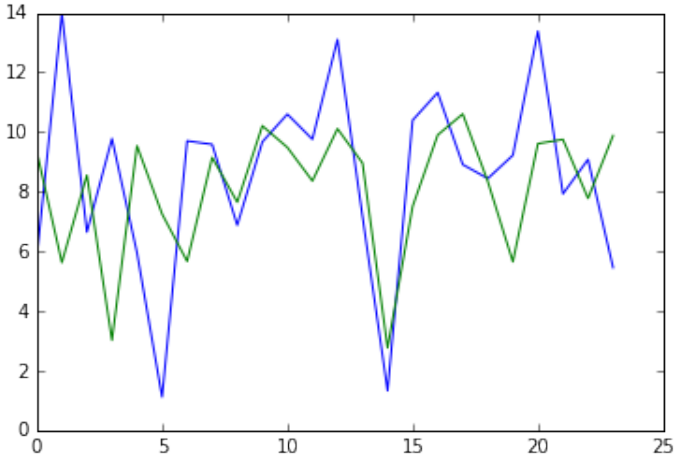
- The presence of prior weighting introduces a bias to the data and any data that does not conform to this bias will be better fitted by finite mixture models
- Requires tuning of an extra hyperparameter

Using a maximum of 80 components, we see if we can obtain a lower absolute mean error while modulating our prior in 10 steps from $10^{-2}$ to $10^{8}$.




We see the lowest error of 2.347834 at a prior value of 10^3.

The predictions for this model are:




As we can see, this model is much more accurate in predicting the large variation in data due to the larger max component count without overfitting due to the presence of the prior. In fact, the graph is largely correct in its peak magnitudes aside from some being slightly shifted by an hour or two on where their actual peak position should be.

Since this is most effective for stationary processes, we will later attempt to augment TS forecasting with seasonal data to account for the seasonality of power readings.

## VII. K-Nearest Neighbors and Recurrent Neural Network Approaches

In order to better represent these temporal relations, we implemented more complex models, focusing on a K-Nearest Neighbors (KNN) model and a Recurrent Neural Network (RNN). Based on the improvements of the EM Model over the baseline model, we hypothesized that a more complex model would be able to better learn the temporal relationships and dependencies inherent in the historic data. Specifically, we noticed that the EM model would underfit outliers of the model and center at the mean predictions too often.

### *A. Iterated Time Series Forecasting Using K-Nearest Neighbors*

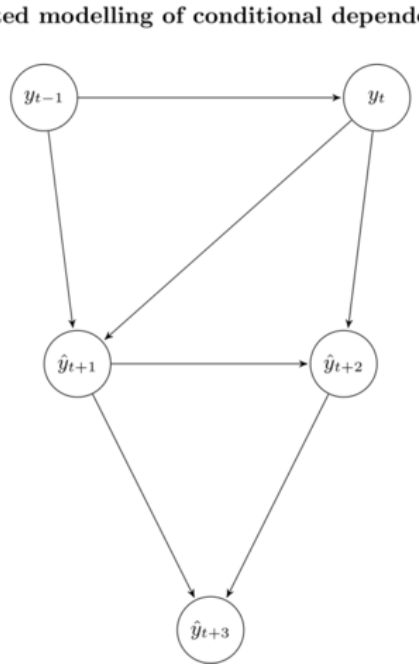
First, let's take a look at non-regressive time series forecasting. Let $y_t$ denote the value of the time series at time point $t$, then we assume that

$$y_{t+1} = f(y_t, ..., y + tn + 1) + \epsilon_t$$

,

for some autoregressive order $n$ and where $_t$ represents some noise at time $t$ and $f$ is an arbitrary and unknown function. The goal is to learn this function $f$ from the data and obtain forecasts for $t + h$, where $h \in 1, ..., H$. Hence, we are interested in predicting the next $H$ data points, not just the $H$th data point, given the history of the time series.

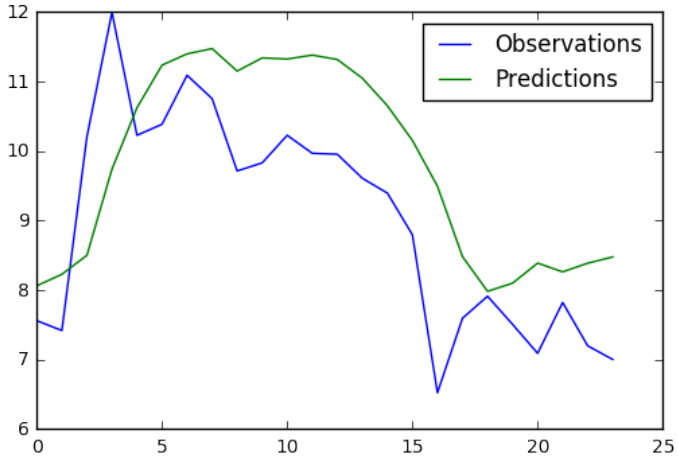
In iterated forecasting, we optimize a model based on a predicting for a window of one step. When calculating a H-step ahead forecast, we iteratively feed the forecasts of the model back in as input for the next prediction. Essentially, what we are trying to approximate and maximize is $[y_{t+1} : y_{t+H} | y_{tn+1} : y_t]$ where $y_{t+1} : y_{t+H} = [y_{t+1}, ..., y_{t+H}] \in R^H$ and $y_{tn+1} : t = [y_{tn+1}, ..., y_t] \in R^H$ where n is the autoregressive window. To visualize this, for a window $n = 2$, the model is visualized as:



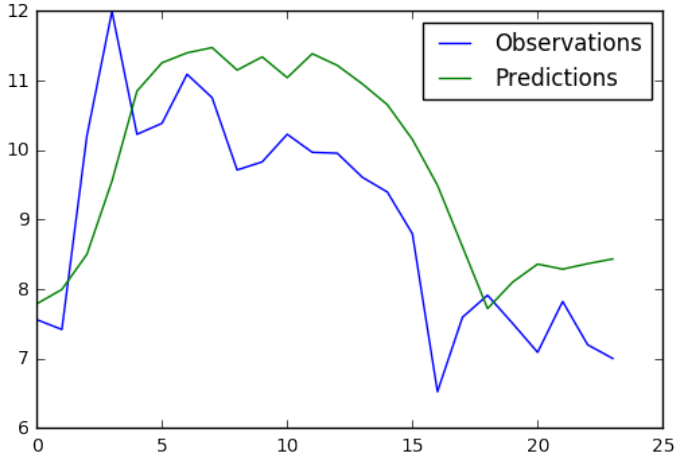
The iterated strategy is still an unbiased estimator of $[y_{t+1} : y_{t+H}|y_{tn+1} : y_t]$ as it preserves the stochastic dependencies of the underlying data. In terms of the bias-variance trade-off, however, the accumulation of error in the individual forecasts causes the model to have high variance. This means that we will get a low performance over longer time horizons H. To keep in parity with our EM predictions, we use 24 autoregressive terms in our historical window and iteratively predict the power readings for 24 future hours. When including the weather data, we append 10 further values containing our weather metrics for the current time to the 24 autoregressive terms.

The K-Nearest Neighbor (KNN) Algorithm is a non-parametric method used for classification and regression. An unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the $k$ training samples nearest to that query point. We use K=10 to look at the 10 most similar days in the past to predict the next day. We hope that this will overcome any averaging that occurs in similar GMM approaches we attempted in the previous section.

Sample day of power readings predicted using only iterated time series forecasting and no weather data:




Sample day of power readings predicted using time series forecasting and weather data:



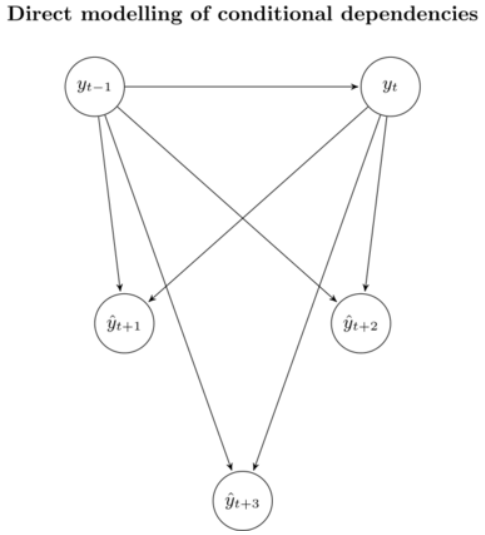

As seen based on the sample data alone, the inclusion of weather data does not improve our iterated time series forecasting much. Firstly, this is because the weather data for only the $t-1$-th term is used and no temporal relationship is found amongst the weather features. Secondly, our errors propagate as we step through iteration of our prediction. We can see from the graphic that the errors in our predicted values will further deviate later predicted values from being correct due to skewed conditionals. We attempt to rectify this by using a multi-output model.

### *B. Multi-Input Multi-Output (MIMO) Time Series Forecasting using Recurrent Neural Networks*

By using a model that trains on an H-dimensional output, we avoid the problem of having our errors propagate from our predictions. We then try to estimate the following function:

$$[y_{t+H}, ..., y_{t+1}] = f(y_t, ..., y_{tn+1}) + \epsilon$$

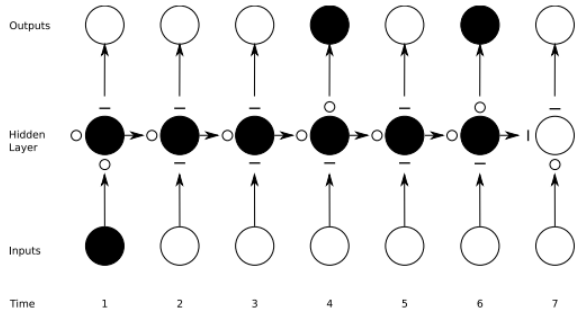
Essentially, our new model of conditionals will look similar to this:



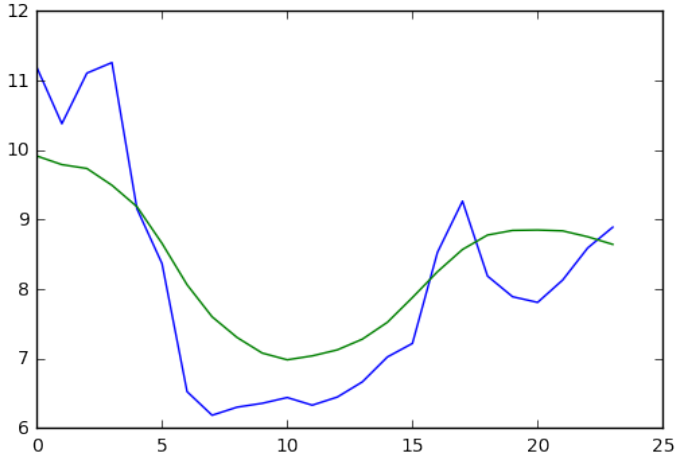

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. For this problem in particular, the Recurrent Neural Network encodes the entire lookback window (the past $H$ hours) and does not assume any conditional dependences between iteratively generated predictions and future predictions. The Recurrent Neural Network should be better able to learn the historic trends of days and not just average out historic trends. Our inspiration for this model is based on previous RNN success in domains with temporal relationships such as Natural Language Processing, Audio, and Speech.

We use Long Short-term Memory (LSTM) neurons to prevent disappearing gradients and for better bias. The baseline architecture we chose was 2 layers of LSTM neurons with hidden layer size of 64 and 256. We have not yet tuned the architecture with any clear deliberation, but have included RNN results to demonstrate the improvement of a more complex model over the baseline and EM models.

The overall structure of the RNN is shown:

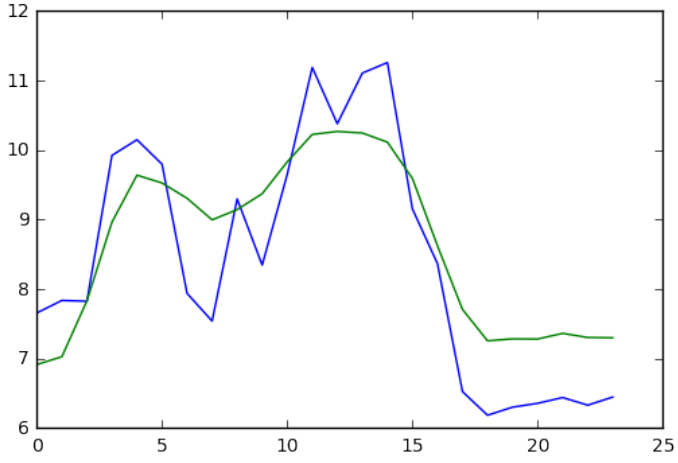



Sample day of power readings predicted using only multi-output forecasting and no weather data:




Absolute Mean Error: 1.10714189049 using a historical window $H$ of size 38 and hidden layer size of 64

Sample day of power readings predicted using only multi-output forecasting and weather data:




Absolute Mean Error: 1.18180182587 using a historical window $H$ of size 28 and a hidden layer size of 256

## VIII. Results

The following results are all measured in the absolute mean errors on 24 hour predictions on a validation set of 591 slices of 24 hour historical windows for all models except for RNN's without and with weather which had 38 hours and 28 hours respectively as tuned hyperparameters:

| Lasso | EM | EM with Dirichlet Prior |
|---|---|---|
| 12.461124 | 2.404167 | 2.347834 |

| K-NN w/o Weather | KNN w/ Weather |
|---|---|
| 1.256917 | 1.258889 |

| RNN w/ Weather | RNN w/o Weather |
|---|---|
| 1.181802 | 1.107142 |

## IX. Conclusion

One of the most surprising results was the high degree of accuracy we were able to obtain for power time series prediction using only the historical power data without any weather information. While each of our progressive methods from the weather based regression approach to EM, KNN, RNN saw successively higher accuracy, the inclusion of weather, even when using temporal relationships in RNNs, saw a slight decrease or no deviation in the accuracy. This may be due to the lack of complete weather forecasting data from forecast.io causing us to remove days that are missing values and causing discontinuities in the time series data that our model erroneously learns.

While the weather series data alone had a high error at an hourly level, we were able to reduce our error drastically using EM. However, one of the problems we faced using EM was the low variation in its predicted values that would not suitably match the peaks in the observed data. To combat this, we used a higher number of components while being cautious about overfitting. Further, we observed that using a Dirichlet Prior allowed us to predict the peaks highly accurately but still had an error above 2 due to the offsets in peak position in the range of a few hours. To combat this, we tried to employ time series forecasting that concretely modeled this temporal relation in time series data in two manners, iteratively (using KNN) and directly (using RNN). The latter was able to predict the power consumption in Sutardja Dai Hall by the main supply fans of its HVAC system with a lowest mean absolute error of 1.107142.

## X. Future Work

We can further extend this work to model the power consumption by the entirety of the HVAC system in the BRITE Sutardja Dai Hall system and then further extend our training set to multiple buildings to see how well it learns for other commercial buildings located in various different districts, housing different types of services, and geospatially well separated. This will allow us to see how well each of these models adapt to the various different types of buildings that may all be present on one power grid.

Furthermore, once we have trained the model to predict the power usage patterns for multiple buildings on a power grid, we can apply intelligent control for peak shaping while having an accurate baseline to understand our deviation.

## ACKNOWLEDGMENT

The authors would like to thank Qie Hu, Professor Claire Tomlin, and the entire Hybrid Systems Laboratory at UC Berkeley for their support throughout this power systems project. We would also like to thank Professor Martin Wainwright and the CS281A course staff for an incredible learning experience this semester.

## REFERENCES

[1] US Energy Information Administration "The Annual Energy Outlook 2013" In *Tec. Rep. '13*

[2] J. Zhang, A. Florita, B.-M. Hodge, S. Lu, H. F. Hamann, V. Banunarayanan, and A. M. Brockway A Suite of Metrics for Assessing the Performance of Solar Power Forecasting In *Solar Energy*

[3] N. Li, J. Kwak, B. Becerik-Gerber, M. Tambe Predicting HVAC Energy Consumption in Commercial Buildings Using Multiagent Systems

[4] B. Dong, C. Cao, E.L. Siew Applying Support Vector Machines to Predict Building Energy Consumption in Tropical Regions In Energy and Buildings '05

[5] E. Eirola, A. Lendasse Gaussian Mixture Models for Time Series Modelling, Forecasting, and Interpolation

[6] Q. Hu, F. Oldewurtel. Model Identification of Commercial Building HVAC Systems during Regular Operation - Empirical Results and Challenges In *ACC '16*

[7] A. Aswani, N. Master, J. Taneja, A. Kriouskov, D. Culler, C. Tomlin. Energy-Efficient Building HVAC Control Using Hybrid System LBMPC In *NMPC '12*

[8] A. Aswani, N. Master, J. Taneja, V. Smith, A. Kriouskov, D. Culler, C. Tomlin. Identifying Models of HVAC Systems Using Semiparametric Regression In *NMPC '12*