Homework 3

1. *Predicting power demand.* Power utilities need to predict the power demanded by consumers. A possible predictor is an *auto-regressive* (AR) model, which predicts power demands based on a history of power demand data, p_t for $t = 1, \ldots, T$. Here, p_t is the power usage in kilowatt hours (kWh) during time interval t.

   The AR predictor uses standardized power data z_t to predict future power demands as an affine function of historical power data. In particular, we set

   $$x^i = (1, z_{i+(h-1)}, \ldots, z_i) \qquad y^i = z_{h+i}, \qquad i = 1, \ldots, T-h,$$

   and predict future standardized power demands $z_{h+i} = y^i \approx \theta^T x^i$. You will select parameter vector $\theta \in \mathbf{R}^{h+1}$ with the following methods.

   (a) Ridge regression

   (b) LASSO regression

   The data in `power_demand_data.json` contains `p`, the hourly electric power demands for California between July 2015 and December 2017 (the times corresponding to the power demands can be found in `dates`).

   Although more sophisticated validation techniques are possible, use the last 4,800 data points (last 200 days' worth of data) to train the model parameters, and validate the regularization weight λ using the remaining records, with $h = 192$ (corresponding to 8 days' data). Examine the validation RMSE for $0.1 \leq \lambda \leq 10$ (possibly with 10 logarithmically spaced points) and choose your optimal λ as the largest λ among the ones with the validation RMSE not exceeding 2% of the minimum validation RMSE.

   Provide plots of validation RMSE versus λ, and plots of the components of your optimal θ. What are the largest absolute values of θ? Make up a plausible story about why these coefficients are the largest ones.

   Using the optimal parameter vectors you obtained from above, predict the future power demand for the next 7 days ("future" of course means the hours after the last data point...). When in the next week do you anticipate that the peak power consumption in California occurs?

   Note. You are welcome to develop your custom prox-gradient code for problem (b). Otherwise, using Julia 0.6 version with the `Convex.jl` package and `ECOS` solver is recommended, though the `ECOS` solver is not based on the prox-gradient method.