Homework #1

Due: February 13, 2026 at 11:59 PM

Regression

Introduction

This homework is on different three different forms of regression: nearest neighbors regression, kernelized regression, and linear regression. We will discuss implementation and examine their tradeoffs by implementing them on the same dataset, which consists of temperature over the past 800,000 years taken from ice core samples.

The folder `data` contains the data you will use for this problem. There are two files:

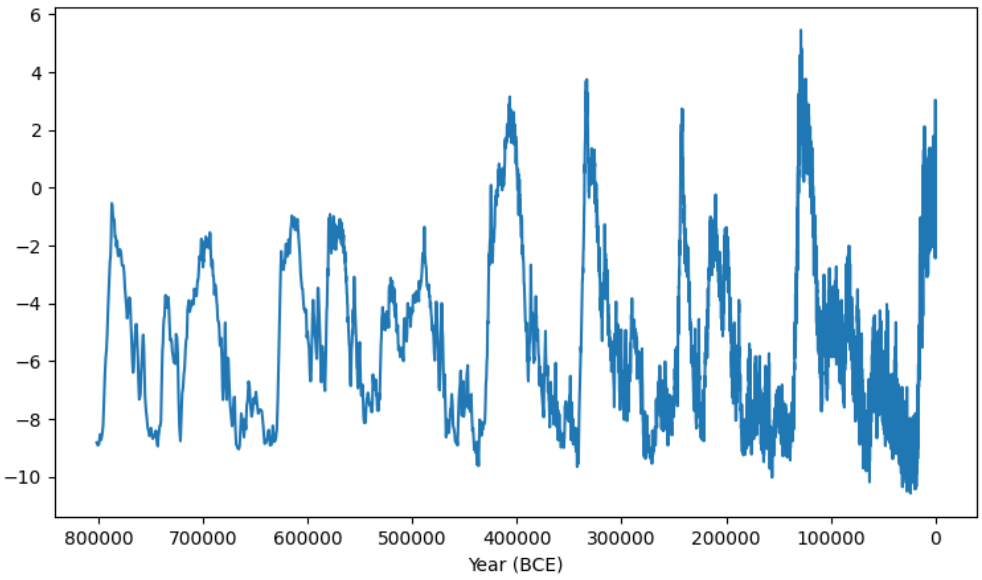
- `earth_temperature_sampled_train.csv`
- `earth_temperature_sampled_test.csv`

Each has two columns. The first column is the age of the ice core sample. The second column is the approximate difference in a year's temperature (K) from the average temperature of the 1,000 years preceding it. The temperatures were retrieved from ice cores in Antarctica (Jouzel et al. 2007)[1].

The following is a snippet of the data file:

```
# Age, Temperature
399946,0.51
409980,1.57
```

And this is a visualization of the full dataset:



Due to the large magnitude of the years, we will work in terms of thousands of years BCE in these problems. This is taken care of for you in the provided notebook.

[1] Retrieved from https://www.ncei.noaa.gov/pub/data/paleo/icecore/antarctica/epica_domec/edc3deuttemp2007.txt
Jouzel, J., Masson-Delmotte, V., Cattani, O., Dreyfus, G., Falourd, S., Hoffmann, G., ... Wolff, E. W. (2007). Orbital and Millennial Antarctic Climate Variability over the Past 800,000 Years. *Science, 317*(5839), 793–796. doi:10.1126/science.1141038

Resources and Submission Instructions

Please type your solutions after the corresponding problems using this LaTeX template, and start each main problem on a new page.

Please submit the writeup PDF to the Gradescope assignment 'HW1'. Remember to assign pages for each question. **You must include any plots in your writeup PDF.** . Please submit your LaTeXfile and code files to the Gradescope assignment 'HW1 - Supplemental.' The supplemental files will only be checked in special cases, e.g. honor code issues, etc. Your files should be named in the same way as we provide them in the repository, e.g. `hw1.pdf`, etc.

If you find that you are having trouble with the first couple problems, it may be helpful to refer to Section 0 notes and review some linear algebra and matrix calculus.

Problem 1 (kNN and Kernels, 35pts)

You will now implement two non-parametric regressions to model temperatures over time.

Make sure to include all required plots in your PDF. Passing all test cases does not guarantee that your solution is correct, and we encourage you to write your own.

1. Recall that kNN uses a predictor of the form

$$f(x^*) = \frac{1}{k} \sum_n y_n \mathbb{I}(x_n \texttt{ is one of k-closest to } x^*),$$

where $\mathbb{I}$ is an indicator variable.

   (a) The kNN implementation **has been provided for you** in the notebook. Run the cells to plot the results for $k = \{1, 3, N-1\}$, where N is the size of the dataset. Describe how the fits change with k. Please include your plot in your solution PDF.

   (b) Now, we will evaluate the quality of each model *quantitatively* by computing the error on the provided test set. Write Python code to compute the test MSE for each value of k. Report the values here. Which solution has the lowest MSE?

2. *Kernel-based regression* techniques are another form of non-parametric regression. Consider a kernel-based regressor of the form

$$f_\tau(x^*) = \frac{\sum_n K_\tau(x_n, x^*) y_n}{\sum_n K_\tau(x_n, x^*)}$$

where $\mathcal{D}_{\texttt{train}} = \{(x_n, y_n)\}_{n=1}^N$ are the training data points, and x^* is the point for which you want to make the prediction. The kernel $K_\tau(x, x')$ is a function that defines the similarity between two inputs x and x'. A popular choice of kernel is a function that decays as the distance between the two points increases, such as

$$K_\tau(x, x') = \exp\left(-\frac{(x - x')^2}{\tau}\right)$$

where τ represents the square of the lengthscale (a scalar value that dictates how quickly the kernel decays).

   (a) First, implement the `kernel_regressor` function in the notebook, and plot your model for years in the range 800,000 BC to 400,000 BC at 1000 year intervals for the following three values of τ: $1, 50, 2500$. Since we're working in terms of thousands of years, this means you should plot $(x, f_\tau(x))$ for $x = 400, 401, \ldots, 800$. **In no more than 10 lines**, describe how the fits change with τ. Please include your plot in your solution PDF.

   (b) Denote the test set as $\mathcal{D}_{\texttt{test}} = \{(x'_m, y'_m)\}_{m=1}^M$. Write down the expression for the MSE of f_τ over the test set as a function of the training set and test set. Your answer may include $\{(x'_m, y'_m)\}_{m=1}^M$, $\{(x_n, y_n)\}_{n=1}^N$, and K_τ, but not f_τ.

   (c) Compute the MSE on the provided test set for the three values of τ. Report the values here. Which model yields the lowest MSE? Conceptually, why is this the case? Why would choosing τ based on $\mathcal{D}_{\texttt{train}}$ rather than $\mathcal{D}_{\texttt{test}}$ be a bad idea?

(d) Describe the time and space complexity of both kernelized regression and kNN with respect to the size of the training set N. How, if at all, does the size of the model—everything that needs to be stored to make predictions—change with the size of the training set N? How, if at all, do the number of computations required to make a prediction for some input x^* change with the size of the training set N?.

(e) What is the exact form of $\lim_{\tau \to 0} f_\tau(x^*)$?

Solution:

1. **kNN Regression**

   (a) **Problem 1.1(a): How fits change with k**

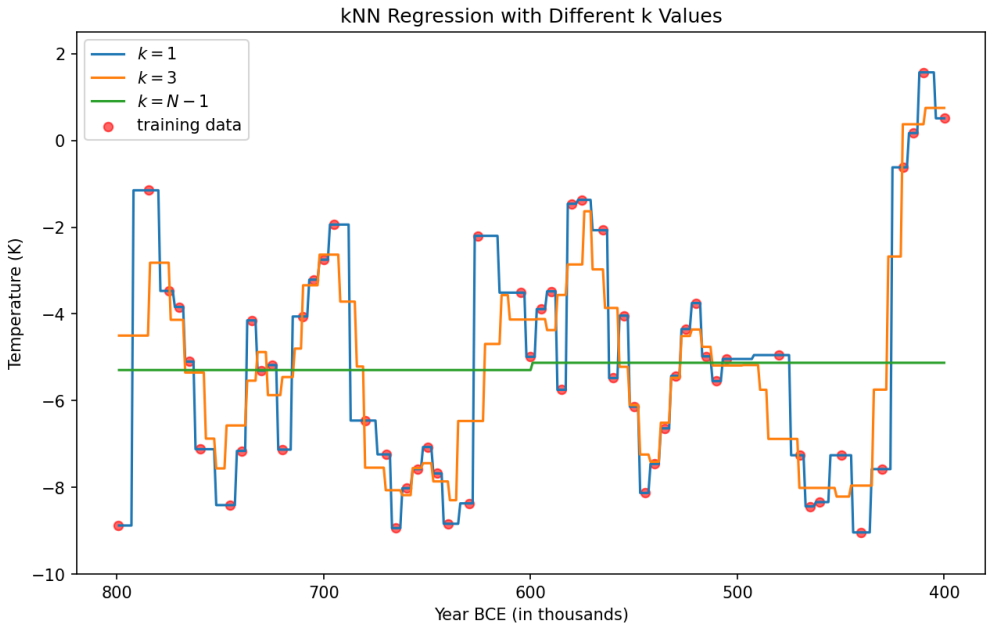


Figure 1: kNN regression for $k = 1, 3, N - 1$ on the temperature dataset.

As k increases:

- $k = 1$: Piecewise constant function passing through every training point. High flexibility, discontinuous jumps.
- $k = 3$: Smoother fit by averaging over 3 neighbors. Reduces discontinuities.
- $k = N - 1$: Nearly constant horizontal line averaging over 56 points. Severe underfitting.

Larger k produces increasingly smooth and rigid fits.

(b) **Problem 1.1(b): Test MSE for kNN models**

Running the code yields the following test MSE values:

k value	Test MSE
$k = 1$	1.7406
$k = 3$	3.8908
$k = N - 1 = 56$	9.5286

The model with $k = 1$ yields the lowest test MSE, suggesting strong local structure in the temperature data. Test error increases monotonically with k, with $k = 56$ showing severe underfitting.

2. **Kernel Regression**

   (a) **Problem 1.2(a): How fits change with τ**

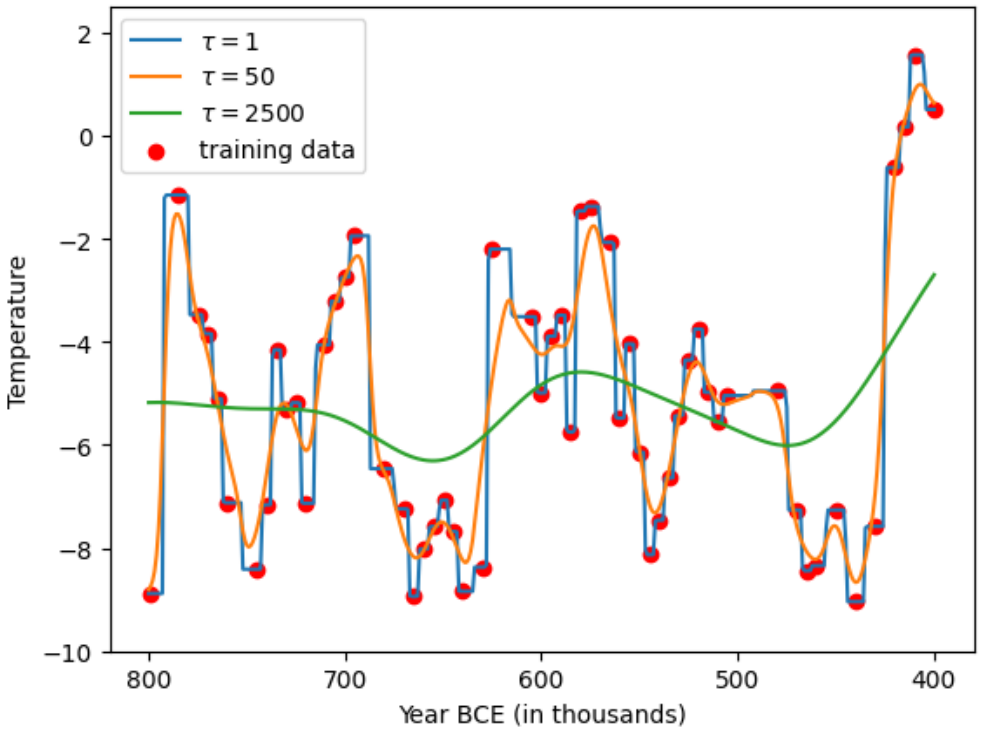


Figure 2: Kernel regression for $\tau = 1, 50, 2500$ on the temperature dataset.

As τ increases:

- $\tau = 1$: Small lengthscale, rapid kernel decay. Very local and wiggly fit, closely following training points.
- $\tau = 50$: Moderate lengthscale. Smooth curve balancing local detail with overall trends.
- $\tau = 2500$: Large lengthscale, slow decay. Extremely smooth fit with broad trends, underfitting.

Larger τ transitions from high variance/low bias to low variance/high bias.

(b) **Problem 1.2(b): MSE expression without using f_τ**

The mean squared error of the kernel regressor over the test set $\mathcal{D}_{\texttt{test}} = \{(x'_m, y'_m)\}_{m=1}^M$ as a function of the training set $\mathcal{D}_{\texttt{train}} = \{(x_n, y_n)\}_{n=1}^N$ and kernel K_τ is:

$$\boxed{\text{MSE} = \frac{1}{M}\sum_{m=1}^{M}\left(y'_m - \frac{\sum_{n=1}^N K_\tau(x_n, x'_m)y_n}{\sum_{n=1}^N K_\tau(x_n, x'_m)}\right)^2}$$

This expression is obtained by substituting the definition of the kernel regressor directly into the MSE formula, eliminating explicit use of f_τ.

(c) **Problem 1.2(c): Test MSE for kernel regression**

The computed test MSE values are:

τ value	Test MSE
$\tau = 1$	1.9473
$\tau = 50$	1.8583
$\tau = 2500$	8.3339

The model with $\tau = 50$ yields the lowest test MSE (1.8583), achieving optimal bias-variance tradeoff. $\tau = 1$ performs nearly as well (1.9473), while $\tau = 2500$ severely underfits (8.3339).

Why choosing τ based on training set is bad: Selecting τ to minimize training error leads to overfitting—we would choose very small τ that memorizes training data. The test set provides an unbiased estimate of generalization. In practice, use validation set or cross-validation for hyperparameter selection.

(d) **Problem 1.2(d): Time and space complexity**

Space: Both kNN and kernel regression store entire training set: $O(N)$.

Time (per prediction): Both require $O(N)$ operations—kNN computes N distances and finds k-nearest, kernel regression computes N kernel evaluations.

Both are non-parametric: model size grows with N.

(e) **Problem 1.2(e): Limit as $\tau \to 0$**

As $\tau \to 0$, the kernel becomes increasingly peaked. For $x^* \neq x_n$, $K_\tau(x_n, x^*) \to 0$. For $x^* = x_i$, $K_\tau(x_i, x^*) = 1$.

Therefore:

$$\lim_{\tau \to 0} f_\tau(x^*) = \begin{cases} y_i & \text{if } x^* = x_i \text{ for some training point } (x_i, y_i) \\ \text{undefined} & \text{if } x^* \neq x_n \text{ for all } n \text{ (form } 0/0) \end{cases}$$

Interpolates training data exactly, unstable between points.

Problem 2 (Geometric Least Squares, 20pts)

Linear regression can be understood using geometric intuition in $\mathbb{R}^n$. The design matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$, with $N > D$, spans a subspace $C(\mathbf{X})$, the column space of $\mathbf{X}$ (referred to in lecture as column span) which is a subspace of $\mathbb{R}^N$. If you wish to review the concept of a column space, consider visiting Section 0 notes.

Fitting by linear regression, sometimes called *ordinary least-squares* (OLS), is just projecting the observation vector $\mathbf{y} \in \mathbb{R}^N$ orthogonally onto that subspace. Lecture 2 slides provide a good graphic to visualize this, see the slide titled "Geometric Interpretation." From lecture, we also learned that $\hat{\mathbf{y}}$ lives in $C(\mathbf{X})$ and the residual $\mathbf{r} = \mathbf{y} - \hat{\mathbf{y}}$ lives in the orthogonal complement.

1. Let $\mathbf{X} \in \mathbb{R}^{N \times D}$ have full column rank d and $\mathbf{y} \in \mathbb{R}^N$. Let the OLS estimator be $\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} X^\top \mathbf{y}$ and the fitted vector $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}^*$. Prove that $\hat{\mathbf{y}}$ is the *orthogonal projection* of y onto $C(\mathbf{X})$. In other words, show that $\hat{\mathbf{y}} \in C(\mathbf{X})$ and $\mathbf{y} - \hat{\mathbf{y}}$ is orthogonal to $C(\mathbf{X})$. *Hint: To show orthogonality, look at the gradient of $\mathcal{L}$, the loss, with respect to* $\mathbf{w}$.

2. Prove that among all vectors in $C(\mathbf{X})$, the fitted vector $\hat{\mathbf{y}}$ minimizes the Euclidean distance to $\mathbf{y}$. In other words, that for every vector $\mathbf{v} \in C(\mathbf{X})$:

$$\|\mathbf{y} - \hat{\mathbf{y}}\|_2^2 \leq \|\mathbf{y} - \mathbf{v}\|_2^2$$

   Looking back at lecture, this is the formal proof of the phenomenon discussed in the image. *Hint: For two vectors, $\mathbf{v}, \mathbf{w}$, if $\mathbf{v}$ is orthogonal to $\mathbf{w}$, denoted as $\mathbf{v} \perp \mathbf{w}$, then $\|\mathbf{v} - \mathbf{w}\|_2^2 = \|\mathbf{v}\|_2^2 + \|\mathbf{w}\|_2^2$ (Pythagorean theorem).*

3. In lecture, we defined the projection matrix, $\mathbf{P} = \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top$, which projects onto the subspace $C(\mathbf{X})$. The matrix $\mathbf{P}$ is often called the *hat matrix* because it maps $\mathbf{y}$ to its fitted values $\hat{\mathbf{y}} = \mathbf{P}\mathbf{y}$, i.e., it "puts a hat" on $\mathbf{y}$. Prove the following properties of $\mathbf{P}$:

   - Symmetry: $\mathbf{P}^\top = \mathbf{P}$
   - Idempotence: $\mathbf{P}^2 = \mathbf{P}$
   - Rank and Trace: $\text{rank}(\mathbf{P}) = d$ and $\text{trace}(\mathbf{P}) = d$.

   Hint: You may use the fact that any idempotent matrix has equal rank and trace. You do not need to prove this, but it may be helpful to think about why this is true.

4. Suppose you fit your model as in Problem 5.1. You observe that the **residual plot** exhibits a clear parabolic (U-shaped) pattern rather than random scatter around zero (as seen in lecture). Give a geometric interpretation of this phenomenon in terms of projection onto the column space of the design matrix. Also, explain how adding a quadratic basis function affects the geometry of the regression problem and the residuals.

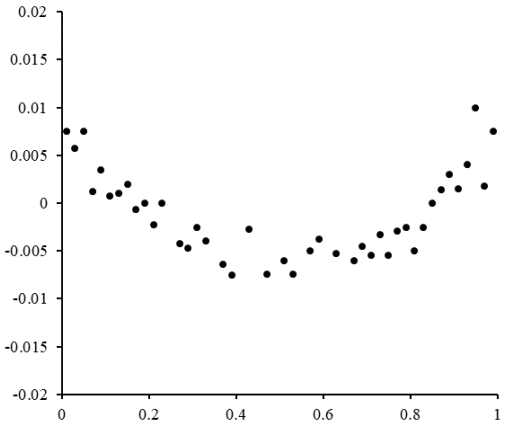


Figure 3: An example residual plot with the input variable x on the horizontal axis and residuals $y - \hat{y}$ on the vertical axis.

Solution:

1. **Problem 2(a): Proof that $\hat{\mathbf{y}}$ is the orthogonal projection**

   We need to show two things: (i) $\hat{\mathbf{y}} \in C(\mathbf{X})$, and (ii) $(\mathbf{y} - \hat{\mathbf{y}}) \perp C(\mathbf{X})$.

   Part (i): $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}^*$ is a linear combination of columns of $\mathbf{X}$, so $\hat{\mathbf{y}} \in C(\mathbf{X})$.

   Part (ii): At the optimum, $\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}^*) = -\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}^*) = \mathbf{0}$. Thus $\mathbf{X}^\top(\mathbf{y} - \hat{\mathbf{y}}) = \mathbf{0}$, meaning the residual is orthogonal to all columns of $\mathbf{X}$, hence orthogonal to $C(\mathbf{X})$.

2. **Problem 2(b): Proof that $\hat{\mathbf{y}}$ minimizes distance**

   For any $\mathbf{v} \in C(\mathbf{X})$, we can write $\mathbf{y} - \mathbf{v} = (\mathbf{y} - \hat{\mathbf{y}}) + (\hat{\mathbf{y}} - \mathbf{v})$. Since $\hat{\mathbf{y}}, \mathbf{v} \in C(\mathbf{X})$, we have $(\hat{\mathbf{y}} - \mathbf{v}) \in C(\mathbf{X})$. From part (a), $(\mathbf{y} - \hat{\mathbf{y}}) \perp C(\mathbf{X})$, so $(\mathbf{y} - \hat{\mathbf{y}}) \perp (\hat{\mathbf{y}} - \mathbf{v})$. By Pythagorean theorem:

   $$\|\mathbf{y} - \mathbf{v}\|_2^2 = \|\mathbf{y} - \hat{\mathbf{y}}\|_2^2 + \|\hat{\mathbf{y}} - \mathbf{v}\|_2^2 \geq \|\mathbf{y} - \hat{\mathbf{y}}\|_2^2$$

   with equality iff $\mathbf{v} = \hat{\mathbf{y}}$.

3. **Problem 2(c): Properties of the projection matrix**

   Let $\mathbf{P} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$.

   Symmetry: We need to show $\mathbf{P}^\top = \mathbf{P}$.

   $$\begin{aligned}
   \mathbf{P}^\top &= \left[\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\right]^\top \\
   &= (\mathbf{X}^\top)^\top \left[(\mathbf{X}^\top\mathbf{X})^{-1}\right]^\top \mathbf{X}^\top \\
   &= \mathbf{X}\left[(\mathbf{X}^\top\mathbf{X})^\top\right]^{-1}\mathbf{X}^\top \\
   &= \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \quad \text{(since } \mathbf{X}^\top\mathbf{X} \text{ is symmetric)} \\
   &= \mathbf{P}
   \end{aligned}$$

   Idempotence: We need to show $\mathbf{P}^2 = \mathbf{P}$.

   $$\begin{aligned}
   \mathbf{P}^2 &= \mathbf{P} \cdot \mathbf{P} \\
   &= \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \cdot \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \\
   &= \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1} \underbrace{(\mathbf{X}^\top\mathbf{X})}_{\text{cancel}}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \\
   &= \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \\
   &= \mathbf{P}
   \end{aligned}$$

   Rank: $\mathbf{P}$ projects onto $C(\mathbf{X})$, a d-dimensional subspace. Therefore, $\text{rank}(\mathbf{P}) = d$.

   Trace:

   $$\begin{aligned}
   \text{trace}(\mathbf{P}) &= \text{trace}\left(\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\right) \\
   &= \text{trace}\left((\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{X}\right) \quad \text{(cyclic property)} \\
   &= \text{trace}(\mathbf{I}_d) = d
   \end{aligned}$$

4. **Problem 2(d): Geometric interpretation of parabolic residual pattern**

   Geometric interpretation: The parabolic residual pattern indicates the true relationship contains a quadratic component not in our model. The column space $C(\mathbf{X})$ lacks the direction for this quadratic pattern, so the residual $\mathbf{y} - \hat{\mathbf{y}}$ contains systematic structure orthogonal to $C(\mathbf{X})$.

   Effect of adding quadratic basis: Adding $\phi(x) = x^2$ expands the column space to $C(\tilde{\mathbf{X}})$, adding a dimension that can capture the quadratic component of $\mathbf{y}$. Residuals then show random scatter instead of systematic structure.

Problem 3 (Basis Regression, 30pts)

We now implement some linear regression models for the temperature. If we just directly use the data as given to us, we would only have a one dimensional input to our model, the year. To create a more expressive linear model, we will introduce basis functions.

Make sure to include all required plots in your PDF.

1. We will first implement the four basis regressions below. Note that we introduce an addition transform f (already into the provided notebook) to address concerns about numerical instabilities.

   (a) $\phi_j(x) = f(x)^j$ for $j = 1, \ldots, 9$. $f(x) = \frac{x}{1.81 \cdot 10^2}$.

   (b) $\phi_j(x) = \exp\left\{-\frac{(f(x) - \mu_j)^2}{5}\right\}$ for $\mu_j = \frac{j+7}{8}$ with $j = 1, \ldots, 9$. $f(x) = \frac{x}{4.00 \cdot 10^2}$.

   (c) $\phi_j(x) = \cos(f(x)/j)$ for $j = 1, \ldots, 9$. $f(x) = \frac{x}{1.81}$.

   (d) $\phi_j(x) = \cos(f(x)/j)$ for $j = 1, \ldots, 49$. $f(x) = \frac{x}{1.81 \cdot 10^{-1}}$. [a]

   *Note: Please make sure to add a bias term for all your basis functions above in your implementation of the `make_basis`.

   Let

   $$\phi(\mathbf{X}) = \begin{bmatrix} \phi(x_1) \\ \phi(x_2) \\ \vdots \\ \phi(x_N) \end{bmatrix} \in \mathbb{R}^{N \times D}.$$

   You will complete the `make_basis` function which must return $\phi(\mathbf{X})$ for each part (a) - (d). You do NOT need to submit this code in your LaTeXwriteup.

   Then, create a plot of the fitted regression line for each basis against a scatter plot of the training data. Boilerplate plotting code is provided in the notebook—you will only need to finish up a part of it. **All you need to include in your writeup for this part are these four plots.**

2. Now we have trained each of our basis regressions. For each basis regression, compute the MSE on the test set. Discuss: do any of the bases seem to overfit? Underfit? Why?

3. Briefly describe what purpose the transforms ϕ serve: why are they helpful?

4. As in Problem 1, describe the space and time complexity of linear regression. How does what is stored to compute predictions change with the size of the training set N and the number of features D? How does the computation needed to compute the prediction for a new input depend on the size of the training set N? How do these complexities compare to those of the kNN and kernelized regressor?

5. Briefly compare and constrast the different regressors: kNN, kernelized regression, and linear regression (with bases). Are some regressions clearly worse than others? Is there one best regression? How would you use the fact that you have these multiple regression functions?

Note: You may be concerned that we are using a different set of inputs $\mathbf{X}$ for each basis (a)-(d), since it could seem as though this prevents us from being able to directly compare the MSE of the models since we are using different data as input. But this is not an issue, since each transformation is considered as

being a part of our model. This contrasts with transformations that cause the variance of the target $\mathbf{y}$ to be different (such as standardization); in these cases the MSE can no longer be directly compared.

[a] For the trigonometric bases (c) and (d), the periodic nature of cosine requires us to transform the data such that the lengthscale is within the periods of each element of our basis.

Solution:

1. **Problem 3.1: Fitted regression plots**

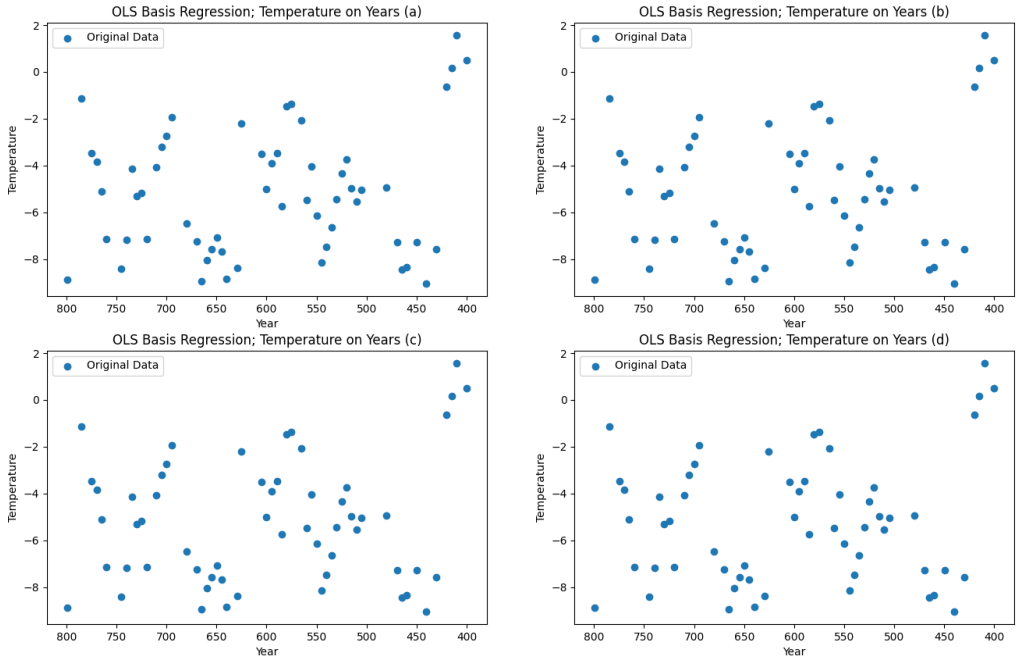


Figure 4: OLS basis regression for four different basis functions: (a) polynomial, (b) RBF/Gaussian, (c) cosine (9 frequencies), (d) cosine (49 frequencies).

The plots show the fitted models for each basis function overlaid on the training data.

2. **Problem 3.2: Test MSE and overfitting/underfitting analysis**

Running the code yields the following test MSE values:

Basis	Test MSE
(a) Polynomial	7.9557
(b) RBF/Gaussian	8.7082
(c) Cosine (9 freq)	5.9670
(d) Cosine (49 freq)	58.9095

Basis (a): MSE = 7.96. Moderate performance, reasonable fit.

Basis (b): MSE = 8.71. Slight underfitting, fixed centers may not align well with data.

Basis (c): MSE = 5.97 (best). Strong periodic structure well-captured by 9 frequencies.

Basis (d): MSE = 58.91. Severe overfitting—49 parameters with only 57 training points provides insufficient constraints.

3. **Problem 3.3: Purpose of basis function transforms ϕ**

Basis functions transform inputs into higher-dimensional feature space:

- **Non-linearity:** Linear in parameters $\mathbf{w}$, non-linear in input x. Enables fitting non-linear patterns.
- **Feature engineering:** Different bases encode different assumptions (polynomial: smooth curves, RBF: local bumps, cosine: periodic).
- **Expressiveness:** Multiple features increase capacity to fit complex patterns.
- **Numerical stability:** Scaling $f(x)$ prevents overflow (e.g., $(x/181)^9$ vs x^9).

4. **Problem 3.4: Space and time complexity of linear regression**

   Space: Store weight vector $\mathbf{w} \in \mathbb{R}^D$. Space: $O(D)$, independent of N. Parametric model.

   Time (per prediction): Compute $\phi(x^*)$ and $\hat{y} = \mathbf{w}^\top \phi(x^*)$: $O(D)$, independent of N.

   Time (training): Computing $\mathbf{w}^* = (\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{y}$: $O(ND^2 + D^3)$.

   Comparison with kNN and kernel regression:

Method	Model Size	Prediction Time	Training Time
kNN	$O(N)$	$O(N)$	$O(1)$
Kernel Regression	$O(N)$	$O(N)$	$O(1)$
Linear Regression	$O(D)$	$O(D)$	$O(ND^2 + D^3)$

   Key differences:

   - **kNN/Kernel** are non-parametric: model grows with data, fast training, slow prediction.
   - **Linear regression** is parametric: fixed model size, upfront training cost, fast prediction.
   - For large test sets or deployment, linear regression is vastly more efficient ($O(D)$ vs $O(N)$ per prediction).
   - For small datasets or exploratory work, kNN/kernel require no training and are simpler to use.

5. **Problem 3.5: Comparison of kNN, kernel regression, and linear regression**

   kNN: Simple, no training, non-parametric. Slow predictions ($O(N)$), discontinuous. Best for small datasets.

   Kernel: Smooth predictions, non-parametric, no training. Slow predictions ($O(N)$), requires tuning τ. Best for smoothness and flexibility.

   Linear with bases: Fast predictions ($O(D)$), compact, closed-form solution. Requires basis selection, can overfit. Best for deployment and large test sets.

   No single best method. Choice depends on dataset size, structure (periodic vs local), deployment needs, and complexity constraints. Use kNN/kernel for exploration, then select appropriate basis for linear regression based on insights. For this dataset, basis (c) with 9 cosine features performs best.

Problem 4 (Probablistic View of Regression and Regularization, 30pts)

Finally, we will explore an alternative view of linear regression to what was introduced in lecture. This view will be probabilistic. We will also introduce Bayesian regression under this probabilistic view. We will explore its connection to regularization for linear models, and then fit a regularized model to the temperature data. The probabilistic interpretation of linear regression is explored in more detail in the course notes under section 2.6.2, but we have also tried to make this question self-contained with all necessary content.

Recall that linear regression involves having N labeled data points, say, $(\mathbf{x}_n, y_n)$ for $n \in \{1, \ldots, N\}$. A probabilistic view of the linear regression problem supposes that the data actually came from a probabilistic model:

$$y_n = \mathbf{w}^\top \mathbf{x}_n + \epsilon_n, \quad \epsilon_n \sim \mathcal{N}(0, \sigma^2).$$

That is, we assume that there exists a set of coefficients $\mathbf{w}$ such that given data $\mathbf{x}_n$, the corresponding y_n results from taking the $\mathbf{w}^\top \mathbf{x}_n$ and adding some random noise ϵ_n. Here, we assume the noise is normally distributed with known mean and variance. The introduction of noise into the model accounts for the possibility of scatter, i.e., when the data does not literally follow a perfect line. It is shown in the aforementioned section of the course notes that under this probabilistic model, the data likelihood $p(\mathbf{y}|\mathbf{w}, \mathbf{X})$ is maximized by $\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$, which, as we already saw in class, also minimizes the squared error. So, amazingly, the probabilistic view of regression leads to the view we saw in lecture, where we are trying to minimize a prediction error.

Now, Bayesian regression takes this probablistic view a step further. You may recall that Bayesian statistics involves choosing a prior distribution for the parameters, here $\mathbf{w}$, based on our prior beliefs. So, in Bayesian regression, we additionally assume the weights are distributed $p(\mathbf{w})$ and fit the weights $\mathbf{w}$ by maximizing the posterior likelihood

$$p(\mathbf{w}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X}) p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}.$$

Note that since we maximize with respect to $\mathbf{w}$, it suffices to just maximize the numerator, while the denominator term does not need to be computed.

1. Suppose $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{\lambda}\mathbf{I})$. Show that maximizing the posterior likelihood is equivalent to minimizing the loss function

   $$\mathcal{L}_{ridge}(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 + \frac{\lambda}{2}||\mathbf{w}||_2^2.$$

   For those who are familiar, note that minimizing $\mathcal{L}_{ridge}(\mathbf{w})$ is exactly what regression with ridge regularization does.

   Hint: You don't need to explicitly solve for the form of the maximizer/minimizer to show that the optimization problems are equivalent.

2. Solve for the value of $\mathbf{w}$ that minimizes $\mathcal{L}_{ridge}(\mathbf{w})$.

3. The Laplace distribution has the PDF

   $$L(a, b) = \frac{1}{2b} \exp\left(-\frac{|x - a|}{b}\right)$$

Show that if all $w_d \sim L\left(0, \frac{2\sigma^2}{\lambda}\right)$, maximizing the posterior likelihood is equivalent to minimizing the loss function

$$\mathcal{L}_{lasso}(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 + \frac{\lambda}{2}||\mathbf{w}||_1.$$

For those who are familiar, note that minimizing $\mathcal{L}_{lasso}(\mathbf{w})$ is exactly what regression with LASSO regularization does.

4. The LASSO estimator is the value of $\mathbf{w}$ that minimizes $\mathcal{L}_{lasso}(\mathbf{w})$? It is very useful in certain real-world scenarios. Why is there no general closed form for the LASSO estimator?

5. Since there is no general closed form for the LASSO estimator $\mathbf{w}$, we use numerical methods for estimating $\mathbf{w}$. One approach is to use *coordinate descent*, which works as follows:

   (a) Initialize $\mathbf{w} = \mathbf{w}_0$.

   (b) For each $d = 1, \ldots, D$ do the following 2 steps consecutively:

      i. Compute $\rho_d = \tilde{\mathbf{x}}_d^\top(\mathbf{y} - (\mathbf{X}\mathbf{w} - w_d\tilde{\mathbf{x}}_d))$. We define $\tilde{\mathbf{x}}_d$ as the d-th column of $\mathbf{X}$.

      ii. If $d = 1$, set $w_1 = \frac{\rho_1}{||\tilde{\mathbf{x}}_1||_2^2}$. Otherwise if $d \neq 1$, compute $w_d = \frac{\text{sign}(\rho_d)\max\left\{|\rho_d| - \frac{\lambda}{2}, 0\right\}}{||\tilde{\mathbf{x}}_d||_2^2}$.

   (c) Repeat step (b) until convergence or the maximum number of iterations is reached.

   Implement the `find_lasso_weights` function according to the above algorithm, letting $\mathbf{w}_0$ be a vector of ones and the max number of iterations be 5000. Then, fit models with $\lambda = 1, 10$ to basis (d) from Problem 3 and plot the predictions on the train set. Finally, compute the test MSE's. You will need to do some preprocessing, but a completed helper function for this is already provided. How do the graphs and errors compare to those for the unregularized (i.e., vanilla) basis (d) model?

Solution:

1. **Problem 4.1: Show ridge regularization equals MAP with Gaussian prior**

   We want to show that maximizing the posterior $p(\mathbf{w}|\mathbf{X}, \mathbf{y})$ is equivalent to minimizing $\mathcal{L}_{ridge}(\mathbf{w})$.

   The posterior is:

   $$p(\mathbf{w}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}$$

   Since the denominator doesn't depend on $\mathbf{w}$, maximizing the posterior is equivalent to maximizing the numerator:

   $$\arg\max_{\mathbf{w}} p(\mathbf{w}|\mathbf{X}, \mathbf{y}) = \arg\max_{\mathbf{w}} p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})$$

   Taking the log (which preserves the argmax):

   $$\arg\max_{\mathbf{w}} \log p(\mathbf{y}|\mathbf{w}, \mathbf{X}) + \log p(\mathbf{w})$$

   Likelihood term: Given $y_n = \mathbf{w}^\top \mathbf{x}_n + \epsilon_n$ with $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$, we have:

   $$p(\mathbf{y}|\mathbf{w}, \mathbf{X}) = \prod_{n=1}^{N} \mathcal{N}(y_n|\mathbf{w}^\top \mathbf{x}_n, \sigma^2) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_n - \mathbf{w}^\top \mathbf{x}_n)^2}{2\sigma^2}\right)$$

   Taking the log:

   $$\log p(\mathbf{y}|\mathbf{w}, \mathbf{X}) = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - \mathbf{w}^\top \mathbf{x}_n)^2 = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2$$

   Prior term: Given $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{\lambda}\mathbf{I})$:

   $$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \frac{\sigma^2}{\lambda}\mathbf{I}) = \frac{1}{(2\pi)^{D/2}|\frac{\sigma^2}{\lambda}\mathbf{I}|^{1/2}} \exp\left(-\frac{1}{2}\mathbf{w}^\top(\frac{\sigma^2}{\lambda}\mathbf{I})^{-1}\mathbf{w}\right)$$

   $$= \frac{1}{(2\pi)^{D/2}(\frac{\sigma^2}{\lambda})^{D/2}} \exp\left(-\frac{\lambda}{2\sigma^2}\mathbf{w}^\top\mathbf{w}\right)$$

   Taking the log:

   $$\log p(\mathbf{w}) = -\frac{D}{2}\log(2\pi\frac{\sigma^2}{\lambda}) - \frac{\lambda}{2\sigma^2}||\mathbf{w}||_2^2$$

   Combining:

   $$\log p(\mathbf{y}|\mathbf{w}, \mathbf{X}) + \log p(\mathbf{w}) = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 - \frac{D}{2}\log(2\pi\frac{\sigma^2}{\lambda}) - \frac{\lambda}{2\sigma^2}||\mathbf{w}||_2^2$$

   Dropping constant terms and multiplying by $-\sigma^2$ (which reverses the optimization from max to min):

   $$\arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 + \frac{\lambda}{2}||\mathbf{w}||_2^2 = \arg\min_{\mathbf{w}} \mathcal{L}_{ridge}(\mathbf{w})$$

   Therefore, maximizing the posterior with a Gaussian prior is equivalent to minimizing the ridge loss.

2. **Problem 4.2: Solve for ridge regression weights**

To minimize $\mathcal{L}_{ridge}(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 + \frac{\lambda}{2}||\mathbf{w}||_2^2$, we take the gradient and set it to zero:

$$\nabla_{\mathbf{w}}\mathcal{L}_{ridge} = \nabla_{\mathbf{w}}\left[\frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^\top\mathbf{w}\right]$$

$$= -\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) + \lambda\mathbf{w} = -\mathbf{X}^\top\mathbf{y} + \mathbf{X}^\top\mathbf{X}\mathbf{w} + \lambda\mathbf{w}$$

Setting to zero:

$$-\mathbf{X}^\top\mathbf{y} + (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})\mathbf{w} = \mathbf{0}$$

Solving for $\mathbf{w}$:

$$\boxed{\mathbf{w}^*_{ridge} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}}$$

3. **Problem 4.3: Show LASSO equals MAP with Laplace prior**

Given that each w_d is independently distributed as $w_d \sim L(0, \frac{2\sigma^2}{\lambda})$, we have:

$$p(w_d) = \frac{1}{2 \cdot \frac{2\sigma^2}{\lambda}} \exp\left(-\frac{|w_d|}{\frac{2\sigma^2}{\lambda}}\right) = \frac{\lambda}{4\sigma^2}\exp\left(-\frac{\lambda|w_d|}{2\sigma^2}\right)$$

The prior for $\mathbf{w}$ is:

$$p(\mathbf{w}) = \prod_{d=1}^{D} p(w_d) = \prod_{d=1}^{D}\frac{\lambda}{4\sigma^2}\exp\left(-\frac{\lambda|w_d|}{2\sigma^2}\right) = \left(\frac{\lambda}{4\sigma^2}\right)^D \exp\left(-\frac{\lambda}{2\sigma^2}\sum_{d=1}^{D}|w_d|\right)$$

Taking the log:

$$\log p(\mathbf{w}) = D\log\left(\frac{\lambda}{4\sigma^2}\right) - \frac{\lambda}{2\sigma^2}||\mathbf{w}||_1$$

The log posterior is:

$$\begin{aligned}\log p(\mathbf{w}|\mathbf{X}, \mathbf{y}) &\propto \log p(\mathbf{y}|\mathbf{w}, \mathbf{X}) + \log p(\mathbf{w}) \\ &= -\frac{1}{2\sigma^2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 - \frac{\lambda}{2\sigma^2}||\mathbf{w}||_1 + \text{const}\end{aligned}$$

Maximizing this is equivalent to minimizing (multiply by $-\sigma^2$ and drop constants):

$$\mathcal{L}_{lasso}(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 + \frac{\lambda}{2}||\mathbf{w}||_1$$

Therefore, maximizing the posterior with a Laplace prior is equivalent to minimizing the LASSO loss.

4. **Problem 4.4: Why no closed form for LASSO**

The ℓ_1 norm $||\mathbf{w}||_1 = \sum_{d=1}^{D}|w_d|$ is not differentiable at $w_d = 0$. The absolute value function has a "kink" at zero where left and right derivatives differ. This prevents setting $\nabla_{\mathbf{w}}\mathcal{L}_{lasso} = \mathbf{0}$ to obtain closed-form solution. In contrast, ridge uses ℓ_2 norm which is everywhere differentiable. The ℓ_1 penalty induces sparsity (drives weights to zero) but requires iterative optimization.

5. Problem 4.5: LASSO implementation and results

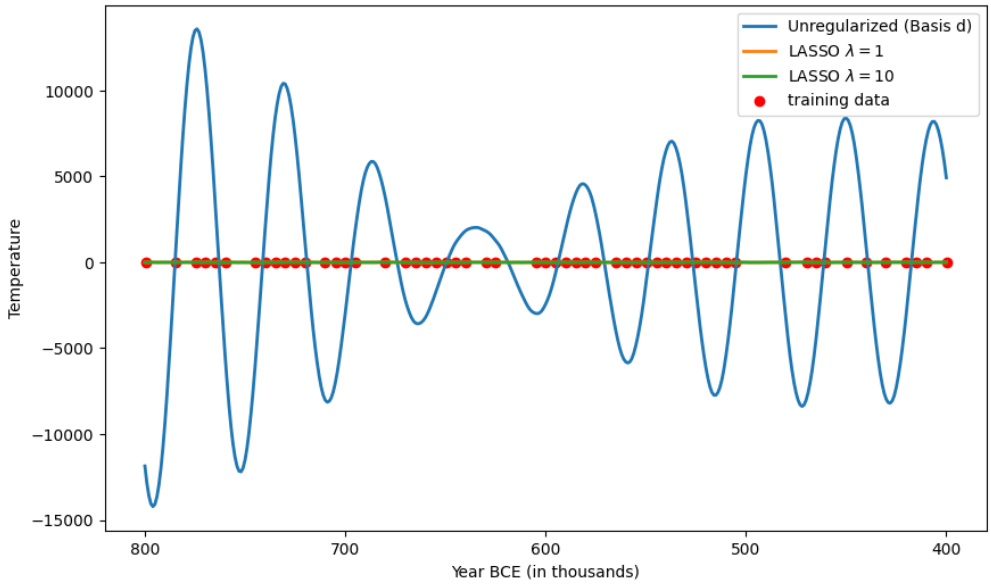


Figure 5: Comparison of unregularized basis (d) regression with LASSO regression for $\lambda = 1$ and $\lambda = 10$.

Test MSE results:

Model	Test MSE
Unregularized (basis d)	377,718,172
LASSO $\lambda = 1$	30.06
LASSO $\lambda = 10$	15.62

The unregularized basis (d) model catastrophically overfits, producing wild oscillations (ranging from $-15{,}000$ to $+12{,}000$) and test MSE over 377 million. LASSO regularization dramatically improves performance: $\lambda = 1$ reduces MSE to 30.06, and $\lambda = 10$ achieves best performance at MSE=15.62 by driving many weights to zero. This demonstrates LASSO's effectiveness at preventing overfitting through sparsity-inducing regularization.

Name:
Collaborators and Resources: