1 Introduction

We focus on the use of Bayesian models and methods for subseasonal temperature forecasting in the western United States. Subseasonal forecasts, which refer to predictions 2-6 weeks ahead, of weather, temperature and other climate-related quantities are of importance to a variety of stakeholders - from private companies in the agricultural sector to governing bodies hoping to more effectively plan the allocation of water resources [1, 2, 3, 4]. In comparison to short and medium term forecasts (1-14 days) and long term forecasts (> 6 weeks), there exists a relative paucity of effective methods for subseasonal forecasting.

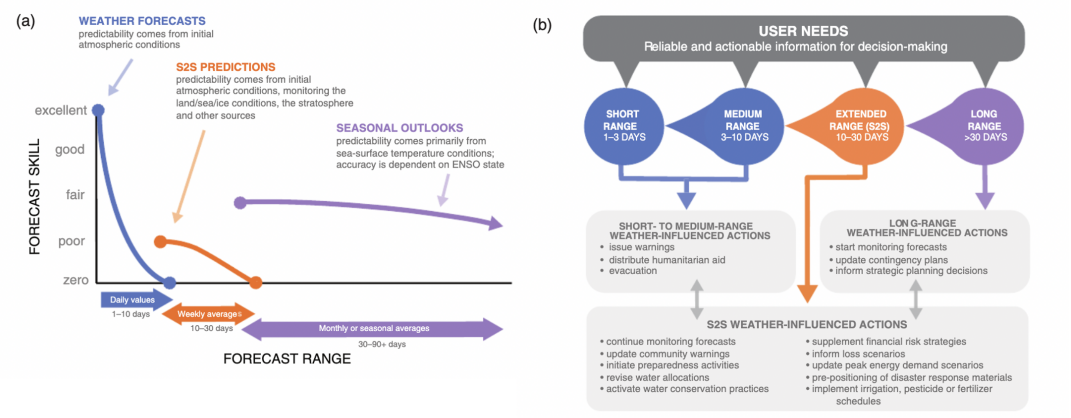


Figure 1: Figure taken from [1]. Subseasonal-to-seasonal (S2S) roughly corresponds to the class of forecasts referred to as subseasonal in this paper. **(a)** shows the forecast quality, referred to as skill, as a function of time from the current day. In different regimes, predictability comes from different sources of information. **(b)** shows the relevant stakeholders for each forecast range.

Figure 1 draws a comparison between various forecasting horizons. The comparison (which is qualitative and not based on numerical experiments/studies) demonstrates the broad inefficiencies in subseasonal forecasts, and sheds some light on why shorter and longer horizon forecasts are more accurate. Figure 1 additionally provides insight into what types of actions and decisions may be taken on the basis of the various types of forecasts.

In this project we apply a number of Bayesian methods and techniques to the subseasonal temperature forecasting task. Given constraints on time and computational resources, a small descriptive dataset is identified and used for modeling purposes. We test the use of Gaussian process regression (GPR), Bayesian structural time series (BSTS) and dynamic linear models (DLM) for modeling and forecasting temperature time-series. We find that prediction results are comparable to those in [2].

1.1 Subseasonal Climate Forecast Rodeo

The Subseasonal Climate Forecast Rodeo [5] was a competition administered by the US Bureau of Reclamation from 2017-2018, with the goal of improving subseasonal (2-6 week horizon) forecasts of temperature (in ^{o}C) and precipitation (in mm) in the western United States, at a set of regions of interest between latitudes 25N to 50N and longitudes 125W to 93W totalling 514 grid points. Competitors were judged based on the "skill" of their temperature and precipitation forecasts. The skill metric represents the ability of a forecasting method to predict temperature/precipitation anomalies. For example, given a temperature measurement $T_{m,d,y,lat,lon}$ or prediction $\hat{T}_{m,d,y,lat,lon}$, on a given month m, day d, and year y for a specific location (lat, lon), the anomaly is given by the deviation of that temperature from the climatology: $a_{m,d,y,lat,lon} = T_{m,d,y,lat,lon} - \text{clim}_{m,d,lat,lon}$. The climatology is defined as the long-term average of the measured temperature over all month-day combinations between 1981 and 2010: $\text{clim}_{m,d,lat,lon} = \frac{1}{30}\sum_{1981 \leq y \leq 2010} T_{m,d,y,lat,lon}$. Given a vector $\hat{\mathbf{a}}$ of these predicted anomalies and the corresponding vector $\mathbf{a}$ of true anomalies over all the grid points in the contest such that $\hat{\mathbf{a}} \in \mathbb{R}^{514}$ and $\mathbf{a} \in \mathbb{R}^{514}$, the skill is then defined as the cosine similarity:

$$\text{skill}(\hat{\mathbf{a}}, \mathbf{a}) = \frac{\hat{\mathbf{a}} \cdot \mathbf{a}}{||\hat{\mathbf{a}}||||\mathbf{a}||} = \cos(\hat{\mathbf{a}}, \mathbf{a}) \tag{1}$$

Although the Subseasonal Rodeo Competition (SRC) focused on predicting anomalies for both temperature and precipitation over the whole western United States, we have focused on predicting just temperature at a single latitude/longitude point for the 3-4 week time horizon as our data analysis goal. This relaxation of the task was made in the interest of being economical with time and computational resources - however in theory the methods discussed in this project should be able to scale to forecasting the full set of quantities required by the competition.

2 Related Work

In this project we apply a few different Bayesian methods for time-series forecasting. One method is to use Gaussian process regression (GPR) for forecasting. Though GPR is a broad technique not explicitly designed for time-series data, some other works have used the technique for forecasting [6, 7]. We also test Bayesian structural time series [8], and the related dynamic linear model [9]. These methods have seen large-scale success in time-series forecasting problems.

A recent work that has significantly impacted the field of time-series modeling and forecasting is Prophet, introduced in [10]. Though we do not test Prophet in this project, the availability of the model as a plug-and-play algorithm that can incorporate analyst/domain knowledge is important to the data analysis community as a whole, and a future line of work could explore the use of Prophet in the SRC forecasting task when expert weather forecasters and analysts are involved.

2.1 Contest entry from Hwang et. al

The approaches and methods detailed in [2] serve as the guiding work for our project. Noting that the contest itself does not provide a centralized dataset for entrants to use, the authors collect and curate a `SubseasonalRodeo`dataset consisting of various environmental metrics, including the contest targets of temperature and precipitation as well as data of sea surface temperature (SST), sea ice concentration (ICEC), multivariate ENSO index (MEI), Madden-Julian oscillation (MJO), relative humidity, pressure, geopotential height, and North American Multi-Modal Ensemble (NMME). Notably, these metrics include raw measurements of meteorological data, indices that are a function of other data, or ensembles of more complex forecasts. The `SubseasonalRodeo`dataset, made freely available by the authors of [2], form the entirety of the data used for our project.

In [2] a non-Bayesian approach was taken, consisting of an ensemble of two regression models. Their first model uses linear regression trained on the full variety of environmental metrics in the `SubseasonalRodeo`dataset on dates local to a desired forecasting date from multiple years. This first method also performs an iterative feature selection procedure hypothesizing that not all features of the data are relevant at a particular time. Their second model performs regression as well but only on features derived from nearest neighbors in historical data of the target variable. They find nearest neighbor historical dates based on the mean skill when anomalies proceeding a date are used to forecast the anomaly on the target date, and take scaled versions of these historical anomalies as regression features. Their ensemble method achieves an average skill of 0.3451 for forecasting temperature 3-4 weeks out and outperforms other contest entrants.

3 Background

3.1 Gaussian Process Regression

Gaussian process regression (GPR) is a non-parametric Bayesian approach to regression. The GPR model assumes that the outputs of a desired target function are given as random variables with a joint normal distribution, with covariances determined by a kernel matrix. The kernel matrix relates input-space distances to output-space covariance - the closer two inputs are in the input space (as measured by a kernel function), the more strongly their respective outputs will covary. Following methods outlined in [11], a predictive posterior distribution for the test outputs y^* can be found in closed form, given past observations X and associated outputs y.

3.2 Bayesian Structural Time Series

Bayesian structural time series (BSTS) [12], are a class of state-space models that are well suited to modelling and forecasting time series data. The structural time series is named because it posits that an observed time-dependent output y_t can be decomposed into a sum of components, which may represent some underlying time varying features such as a global trends or seasonality. These models are endowed with a large degree of interpretability as an analyst can examine each component individually and make adjustments as necessary.

A classic example of such a structural time series is given as follows [8]:

$$y_t = \mu_t + \tau_t + \beta^T x_t + \epsilon_t, \quad \mu_t \quad = \mu_{t-1} + \delta_{t-1} + u_t, \quad \delta_t = \delta_{t-1} + v_t, \quad \tau_t \quad = -\sum_{s=1}^{S-1}, \tau_{t-s} + w_t \tag{2}$$

The components of (2) have simple and intuitive interpretations. The observations y_t consist of the sum of a trend μ_t, a seasonal component τ_t, a regression component $\beta^T x_t$ and a measurement/observation noise given by the Gaussian random variable ϵ_t. The trend level μ_t has a slope of δ_t, and both the trend level and the slope are affected by additive Gaussian noise given by u_t, v_t respectively. The seasonal component τ is constructed so that a set of S τ_i has an expectation of 0 (as w_t is zero-mean Gaussian noise). Thus this model admits an intrinsic variation due to underlying seasonalities and fluctuations in slope, while also allowing external data in the form of regressors x_t to be incorporated. The noise variances $\sigma_\epsilon^2, \sigma_u^2, \sigma_v^2, \sigma_w^2$ and the regression coefficients β form the parameters of the model in (2).

Core to the BSTS approach is the use of Kalman filters and Kalman smoothers to construct posterior distributions for latent variables in the structural model, given observations. In the case of (2), the latent variables include the trend level, slope, seasonal components and the linear regression parameters. Because (2) is fundamentally a linear system in which additive Gaussian noise enters, the optimal posterior predictive distributions of the latent variables can be solved for recursively through the use of the Kalman filter, given the noise variances. The Kalman smoother is then able to improve upon the Kalman filter outputs once the entire set of observatiosn y_t has been observed. Thus the Kalman filter and Kalman smoother operate as a pair of standard forward-backwards message passing algorithms [13, 8].

3.2.1 Variational Inference

One method for approximate posterior inference is variational inference. Here, with y as our given data and θ as our model parameters, we approximate the posterior distribution, $p(\theta|y)$ with another distribution $q(\theta)$. Generally, we posit that $q(\theta)$ is from a family of distributions that are nice or convenient to work with, and we want the member of that family that is closest to our desired posterior based on a metric, commonly the Kullback-Liebler (KL) divergence [14]. The KL divergence can be decomposed into the sum of a constant term as well as the negative evidence lower bound (ELBO). Thus, to minimize the KL divergence we can maximize the ELBO.

For performing variational inference with BSTS, we use Tensorflow Probability (`tfp`) [15]. `tfp` has a `build_factored_surrogate_posterior` function that makes the mean field assumption in that each variable is an independent Gaussian and also utilizes predefined differential operations for various BSTS components that map these normals into the proper support space for that component's parameters. This procedure is similar to that of ADVI [16] and though the `tfp` documentation does not explicitly refer to ADVI, we note that one of ADVI's authors is a core developer of `tfp`. With our differentiable computation graph, we can then pass batches of our data through the model, calculate the negative ELBO, and perform gradient descent (using Adam [17]). `tfp` optimizes the parameters of the normal distributions corresponding to each parameter and we later draw samples from this posterior to use in forecasting.

3.2.2 Monte Carlo Methods

Another method for inference is to use Markov Chain Monte Carlo (MCMC) techniques for posterior approximation. Section 4 of [8] describes an MCMC algorithm for inference in BSTS models that utilizes a Kalman filter/smoother based simulation smoothing algorithm developed by [13]. These algorithms are implemented in the popular R library `bsts` [18], which we utilize.

3.3 Dynamic Linear Model

Another approach that we consider is the dynamic linear model (DLM), which is a Bayesian method for time-series modeling. The nomenclature in the literature regarding time-series methods can be confusing, as many authors consider Bayesian dynamic linear models and Bayesian structural time series to be one and the same [9], while others will primarily pick one term over the other (for example in [8, 12] (2) would be referred to as a BSTS while in [19, 20] it would be referred to as a DLM). For our purposes, we will refer to linear state-space models in which the error and noise covariances are known as dynamic linear models. When these components are unknown and must be learned we refer to the model as a BSTS. In this framing, the DLM approach is a less "fully" Bayesian treatment than the BSTS. This comes with a benefit - methods like variational inference and MCMC are necessary in the BSTS approach because structural parameters such as the variance appear in a non-linear manner [9] - thus making posterior inference more difficult. In contrast, the forward-backward Kalman smoothing algorithm is sufficient for inference in the DLM case. This framing is in line with the DLM implementation provided by `PyDLM` [21]. `PyDLM` is a Python library built for modeling, inference and prediction using Bayesian DLMs. Because the model covariance is assumed known and user provided, computationally expensive techniques like MCMC or VI are not needed and large scale model development & model testing can be performed rapidly.

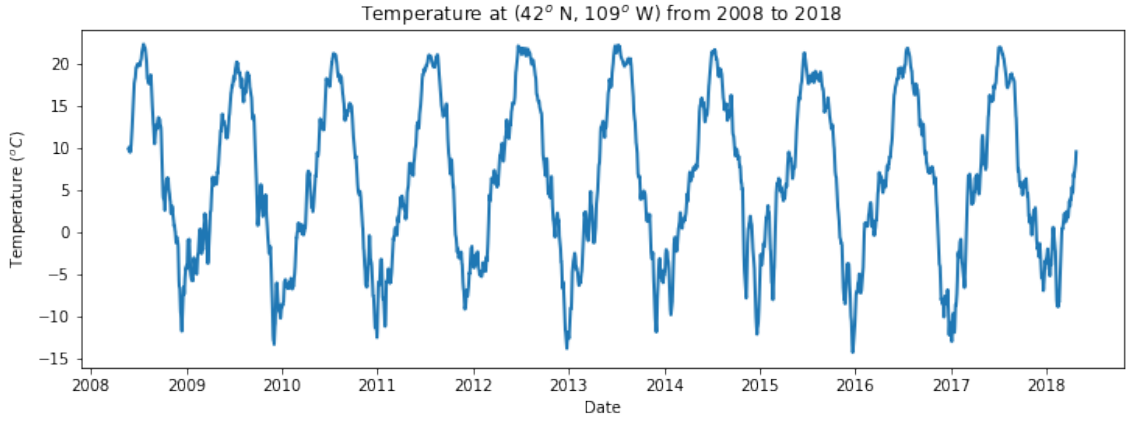


Figure 2: Time-series of temperature at a specified latitude longitude coordinate. Note that temperature generally follows an annual trend that lines with our connotation of seasons.

4 Methods & Results

4.1 Data Processing

In this project we have largely used the dataset provided by the authors of [2]. This dataset consists of atmospheric, temperature and precipitation data collected from a number of publicly available sources, as covered in Section 2.1. In the raw form publicly available online [22], the various dataframes require significant processing to be synthesized into a single dataframe used for their methods. We make the minimal modifications necessary to update the authors' code to be compatible with Python 3. Because the dataset is large - both in the number of variables/features collected and the size & granularity of the geographic area that it covers - we employ a number of basic techniques to reduce the effective dataset size. As with most data-reduction techniques this is likely not a lossless process - in reducing the size of the data we need to deal with, there is some tradeoff between computational tractability and the resulting modeling performance. The main simplification is that we focus only on modeling the temperature of a single geographic point at any given time. This is in contrast with [2] and the Subseasonal Rodeo data analysis goal, which is to predict both temperature and precipitation for all 514 geographic locations. Notably, their method does not jointly predict temperature and precipitation at all geographic locations but instead considers each location and prediction time as an individual regression problem before aggregating for overall performance. By only considering one point, we substantially reduce the dimensionality of the problem. This allows us to maintain a relatively high dimensionality in the other facets of the problem - for example, in having a large set of features for regression - while still ensuring that computations can be performed reasonably fast and within our time & hardware constraints. We also choose to only predict temperature using a 4 week (28 day) forecast horizon - this is again in contrast to [2] and the competition which require separate forecasts of $3-4$ weeks and $5-6$ weeks. The specific point we work with is latitude 42.0^o N and 109.0^o W which corresponds to the southwestern corner of Wyoming. Temperature data from this location is shown in Figure 2.

Given the structural time series model in (2), the observable y_t then corresponds to the temperature of a given coordinate at step t, the regressors consist of the variables collected in the `SubseasonalRodeo`dataset (for comprehensive details on these variables, refer to [2]) and the seasonality period S is a hyperparameter that must be selected for the model.

4.2 Gaussian Process Regression

Model	Skill (avg±std)
GP [Sec 4.2]	0.475±0.597
BSTS [Sec 4.3.1]	0.047±0.725

Table 1: Average skill over 19 subseasonal prediction tasks spaced between 2011 and 2018. Both methods perform better than a skill of zero which would be just predicting the climatology. The Gaussian process model achieves a much higher skill than the structural time series model. Note the large variance for each of these means. Figure 3 plots the histogram of the raw values.

Similar to the analysis done by [11] on Mauna Loa Atmospheric CO_2, we build a Gaussian process model consisting of a combination of standard covariance functions and estimate the parameters of these kernels by minimizing the negative log likelihood of batches of data. The input feature space of our Gaussian process is simply time in years while the target variable is temperature at that time. We note that changes in temperature are likely a result of a range of meteorological processes and not simply a function of time. However, Figure 2 shows that at a given location temperature exhibits a periodic nature as the seasons change.

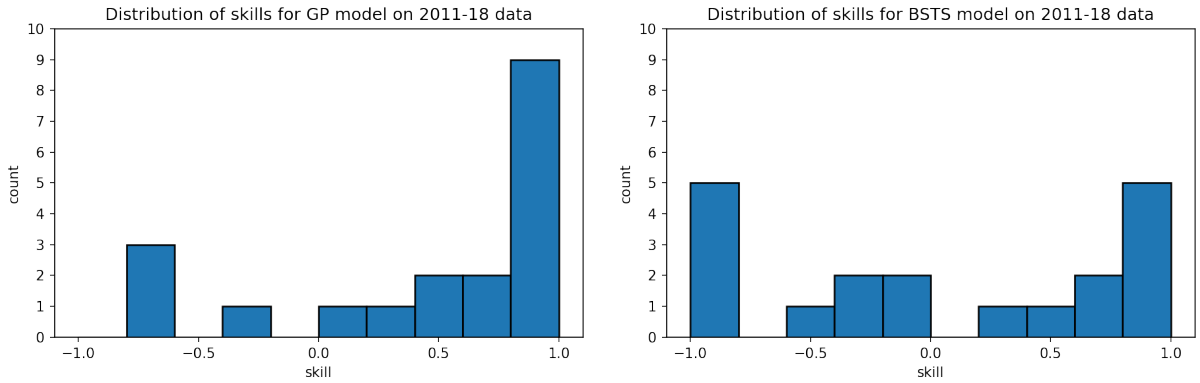


Figure 3: The histogram of skills of two models (GP [Sec 4.2] on the left and BSTS [Sec 4.3.1] on the right) for making 19 subseasonal prediction tasks spaced between 2011 and 2018. Note how the GP model achieves high skill for 9 of the 19 forecasts but also achieves much lower skill on some of the other forecasts. On the other hand, the BSTS model achieves both high and low skills 5 times, whereas other times it achieves scores in between.

Parameters	Value
`smooth_amplitude`	2.550
`smooth_length_scale`	0.022
`periodic_amplitude`	14.111
`periodic_length_scale`	1.869
`periodic_period`	1.001
`periodic_local_length_scale`	319.985
`observation_noise_variance`	0.584

Table 2: Optimized parameters for the Gaussian process example shown in Figure 4a obtained by minimizing the negative log likelihood of batches of data.

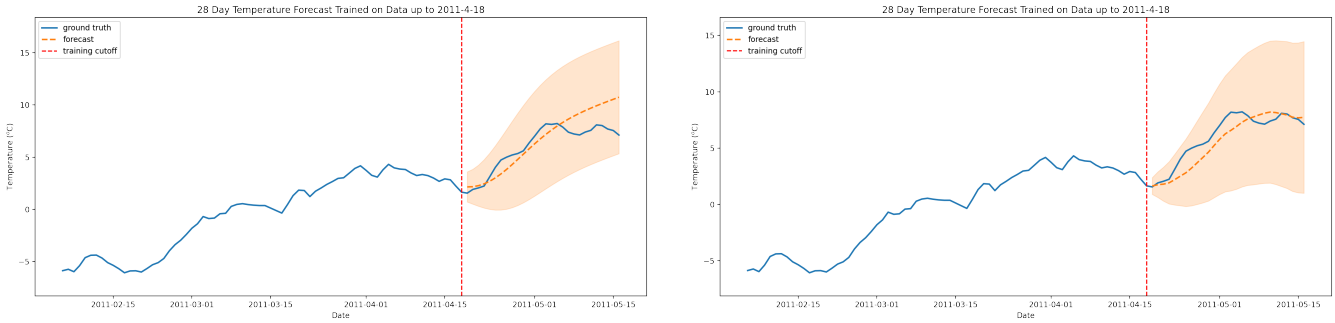
Specifically, we use a squared exponential kernel to capture the long term smooth trend as well as a squared exponential kernel multiplied by a sinusoidal kernel to capture the periodicity as done by [11]. After fitting our data for one specific forecast as shown in Figure 4a, we obtain the kernel parameters as shown in Table 2. We find that the approximated period of the periodic component is close to 1 year, which lines up with our intuitions about the seasonality of temperature. We additionally find that the length scale of the squared exponential it was multiplied with is 320 years, suggesting that this periodicity is quite dependable. The amplitude of this component is also the highest, at 14.1^oC. For the other estimated parameters, we find that the long term smooth trend is actually quite sporadic with a short length scale of 0.02 years and a relatively smaller magnitude at 2.5^oC than the periodic component. Additionally, the observation noise variance is 0.58.

For a larger scale analysis, we evenly space 19 prediction dates across April 18, 2011 and April 18, 2018, deliberately choosing a larger time span to better asses the performance of our model. For each prediction date, we train our model using historical data before the prediction date and then forecast 28 days into the future from the prediction date. For each of these 19 forecasts, we calculate the skill for our subseasonal forecast between days 14 and 28 plot their distribution in Figure 3 and calculate their mean and standard deviation in Table 1. The resulting skill distributions are comparable to those presented in Table 1 of [2]. Although the standard deviations are not presented in [2], we visually estimate from the provided histograms that the standard deviation is about .4. Each forecast took approximately 3 minutes to run on an NVidia RTX 2080 Ti with GPU acceleration from `tfp`.

4.3 Structural Time Series

4.3.1 Variational Inference

As discussed in Section 3.2.1, we use `tfp` for temperature modeling and inference. We define our structural time series to be composed of three components - a smooth seasonal component, a linear regression component, and a noise component. Notably, the smooth seasonal component requires manual specification of a few hyperparameters (a period and a set of frequency multipliers) while another hyperparameter is optimized via gradient descent (the drift scale, which captures how seasonal effects may drift over time). For example, with the temperature data we specify a period of 365 days and expect that there could be a biannual, quarterly, or monthly pattern within the annual pattern, so the multipliers are specified as 2, 4, 12, and 1 respectively. The linear regression component uses the full set of features used for linear regression by [2]. Finally, an observation noise



(a) Gaussian process model

(b) Bayesian structural time series model

Figure 4: The same 28 day forecast using a GP model on the left and BSTS model on the right. On both plots, the dashed orange line is the forecast mean, the shaded orange region represents two standard deviations of uncertainty, the blue line is the ground truth temperature data, and the vertical dotted line represents the cut off date for data used for training and from when the forecast is made.

component captures anything else the model cannot explain. We use the default priors provided by `tfp` for the variables we are optimizing here (the drift scale, linear regression weights, and observation noise variance) which are generally derived from heuristics based on training data or suggestions from the STAN community.

We show an example of a BSTS trained for and performing a forecast in Figure 4b, the same data as with the GP. Notably, the BSTS also includes a linear regression component whereas the GP relied solely on timestamps. The numerical range on the features varied quite significantly so each feature is normalized by the mean and standard deviation across all training points. We drew 50 samples from our posterior and their statistics are shown in Table 3. We find that the features with the highest weight magnitude after training are `wind_hgt_10_2010_1_shift30`, `sst_2010_1_shift30`, and `sst_2010_3_shift30`. The contest submission by [2] also iteratively remove features from regression which we do not implement but could be explored.

Parameter	Mean	Variance
observation_noise_scale	0.091	0.001
annual_drift_scale	0.151	0.03
lr_wind_hgt_10_2010_1_shift30	-3.900	0.957
lr_sst_2010_1_shift30	2.795	1.164
lr_sst_2010_3_shift30	1.183	0.385
lr_sst_2010_2_shift30	-0.844	0.817
lr_tmp2m_shift29_anom	-0.617	0.119
lr_tmp2m_shift58	0.285	0.165
lr_tmp2m_shift29	0.267	0.242
lr_rhum_shift30	0.267	0.142
lr_icec_2010_3_shift30	0.251	0.443

Table 3: Statistics for 50 samples from the posterior for the BSTS example shown in Figure 4b. Parameters beginning with `lr` are linear regression component weights. Only 8 of the 19 linear regression weights with the highest magnitude are shown though all 19 (the same covariates as [2]) are used.

For a larger scale analysis, we follow a similar evaluation procedure here as with the previous section. Notably, for the BSTS training procedure we perform 60 gradient update steps with an Adam optimizer with a learning rate of 0.1. We empirically confirm that within 60 gradient updates the ELBO has plateaued. The skill distribution and statistics are shown on Figure 3 and Table 1, respectively. Each forecast took approximately 5 minutes to run on an NVidia RTX 2080 Ti with GPU acceleration from `tfp`.

4.3.2 Monte Carlo Methods

As discussed, we attempted to use the `bsts` R library for temperature modeling and inference. Although preliminary results appeared to be reasonable, inference proved to be computationally costly. On a commercial laptop (2.3 GHz 8-Core Intel Core i9 processor), fitting a BSTS model (of the form (2)) with $S = 365$, a historical time-series of 1500 days and 430 regressors required nearly 5 minutes per 100 MCMC iterations. Over a set of 5 distinct trials it was found that at least 3000 MCMC iterations were required to achieve a skill of at least .1 on a held-out test set. This indicated that training and fitting a model capable of achieving competitive performance would take many hours. For such an applied project, in which numerous design cycles would be required, this was unacceptable and the `bsts` library was deemed too unwieldy and heavyweight for our purposes. We suspect that the long training times are a result of the use of MCMC for posterior inference. Unlike the variational inference methods discussed, the MCMC routine used by the `bsts` library was not easily accelerated (e.g, by GPU acceleration).

4.4 Dynamic Linear Model

As discussed, the Bayesian dynamic linear model (as described in this project) can be considered a "less-Bayesian" approach to time-series modeling than the more complex BSTS. The upside is that the efficient Kalman filter & smoother are sufficient for posterior inference in the DLM - resulting in a significantly faster and more lightweight algorithm. Given a historical time-series

of 1500 days, 430 regressors and $60 \leq S \leq 365$, we were able to fit a DLM with `PyDLM` in less than a minute on average (with no additional hardware acceleration). This computational boon did not necessarily come at the cost of performance. Although a model fit using `bsts` with default hyperparameters slightly outperformed a model fit using `PyDLM` with default hyperparameters (.19 vs .12 skill on a test set, respectively), the `PyDLM` model was able to infer it's posterior distribution many orders of magnitude faster. This enabled rapid "prototyping", debugging and testing of different model architectures and hyperparameters (such as the seasonality constant S). Furthermore, it enabled higher-confidence and larger-scale evaluations of the fit models: training a model with `PyDLM` and performing 365 evaluations could be accomplished in minutes, while training a model with `bsts` and performing just a handful of evaluations would take many hours.

4.4.1 Data for DLM

Given the increased training and inference speed, we augment the vector of regressors used in the DLM. In addition to the features used in [2] we include another "dimension" of features. This involves adding in temperature and precipitation features (shifted backwards in time appropriately to respect the forward-horizon of the forecasting problem) from the rest of the spatial grid. Because there are 514 total spatial points, adding this many features would lead to a very large state-space. As a result, we downsample the grid to a set of 15 bins in which the meteorological quantities of interest are averaged appropriately. This captures the intuition that temperature and precipitation effects are fundamentally non-local - indicating that data from everywhere on the spatial grid may be helpful in modeling temperature/precipitation fluctuations at a given point. As this non-local data was not used in [2] we hope that using it in our models would lead to improved performance.

4.4.2 Bayesian Optimization

The speed of inference and evaluation afforded by the `PyDLM` models made it possible to perform hyperparameter optimization. Given a DLM composed of a linear trend, a regression component and a seasonality, we optimized the following hyperparameters: the linear trend discount factor, the seasonality discount factor, the linear trend variance, the seasonality variance, and the seasonality constant S. The discount factors are positive numbers between 0 and 1 inclusive (in practice, they are quite close or equal to 1) that represent how fast sequential data in the time-series should be forgotten (or rather, how quickly new data should be incorporated into the model) [20, 21]. While the `PyDLM` library provides a tuning method to optimize the discount factors, this method relies on a first-order gradient descent approach, and only (locally) optimizes the one-step prediction error on the training set [21]. Instead, we use a Gaussian process based Bayesian optimization routine, similar to the approach taken in [23]. The BayesianOptimization Python library [24] is used, along with the Expected Improvement (EI) acquisition function and a Mattern 2.5 kernel. Reasonable bounds for each hyperparameter are chosen (e.g, [.8, 1.0] for a discount factor or [4, 365] for seasonality) and the hyperparameters are normalized to the range [-10, 10] for the Bayesian optimization procedure. The black-box objective to be maximized was the fitted model's average performance (as measured by 28-day temperature forecast skill) on a a set of held-out validation time series. The GP based optimization was run for 250 iterations and the resulting hyperparameter solutions were used in the subsequent DLM tests.

4.4.3 Results

The resulting DLM was evaluated on 6 test series: Table 4 indicates that DLM performs reasonably well. Although a lower

Model	**2017-2018 Skill**	**2016-2017 Skill**	**2015-2016 Skill**	**2014-2015 Skill**	**2013-2014 Skill**	**2012-2013 Skill**
DLM	.38	.28	.50	.37	-0.476	.15
RPT	-.04	.11	-.07	-.15	.19	.04
RND	.11	.16	-.01	-.06	.28	.02

Table 4: Three methods are compared on 6 distinct test sets, each requiring a year of predictions. Each prediction consists of a forecast of the temperature 28 days in the future, given data/features upto and including the current date. The forecast is logged, and the model is "stepped" into the next day and updated data is provided. Using the updated data, the DLM is re-fit in preparation for the next forecast. For each period, predictions are made between April 17th of the current year and April 17th of the next year - leading to a total of 337 predictions. The skill is then calculated as the cosine similarity between the 337 predicted anomalies and the 337 true anomalies for the given year. We benchmark DLM against two other methods - RPT, which forecasts the 28-day temperature as the current temperature, and RND which predicts a random, uniformly sampled anomaly from the interval [-5, 5]. Curiously, RND performs well and even outperforms DLM for the period between 2013 and 2014. This potentially calls into question the evaluation metrics used in this project, as well as the use of cosine similarity as a metric.

mean skill is achieved than GPR, the results are comparable to the baselines and competitor skills achieved in [2]. Moreover, the DLM skill estimates are somewhat higher confidence because each tested year consists of 337 anomaly predictions, instead of the 19 total used in GPR. This is made possible by the rapid inference and prediction speed of the DLM. Figure 5 (left) gives an example plot of the DLM predictions made for the year 2012-2013.

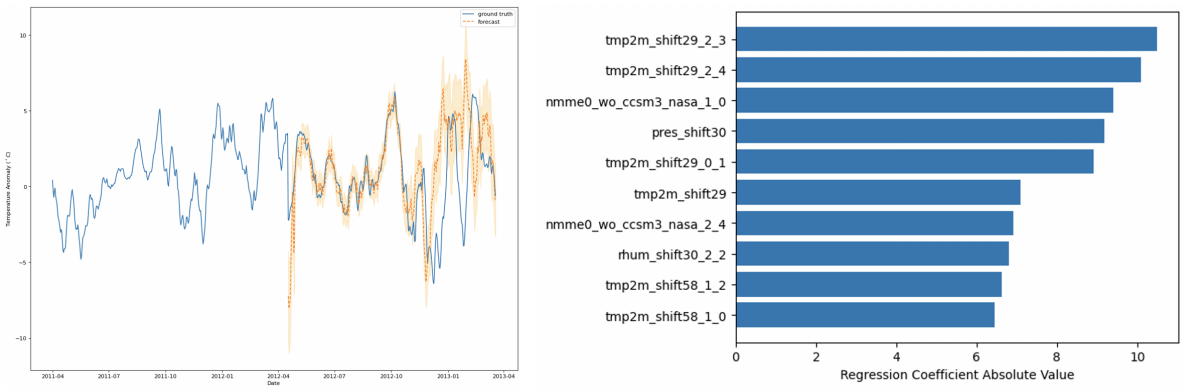


Figure 5: **Left:** A year's worth of 28 day temperature forecasts. Data from 2011-2012 is used for model fitting, and prediction is tested from 2012-2013. Because the DLM can be fit quickly, the model is re-fit to include the new data at each step. When discount factors of less than 1 are used, this generally improves the model as the DLM is able to incorporate new movements and trends more effectively. The two standard-deviation bounds are given by the shaded region. **Right:** An example of the top ten most salient regression variables for the DLM as measured by the magnitude of the regression coefficient. Variables of the form `var_name_i_j` correspond to the average value of that variable over the downsampled bucket with coordinates i, j. In this particular example, the spatial grid was downsampled such that $0 \leq i \leq 2$ and $0 \leq j \leq 4$. For example, the highest weighted regressor, `tmp2m_shift29_2_3`, is the average 29-day stale temperate of the downsampled bucket with coordinates (2,3).

We also inspect the top ten regression variables (as measured by the absolute value of the associated regression coefficient). Figure 5 (right) provides an indication of the types of variables that are weighted in the regression component of the DLM. Interestingly, the local shifted temperature (measured at the location we would like to forecast at) appears to be less prominent in the regression component than some of the non-local shifted temperatures (which are average temperate measurements at different locations). This indicates that the inclusion of non-local temperature data aided in the model fitting and performance, and may be a relative advantage over the method proposed in [2].

5 Discussion

Three Bayesian methods for time-series modeling and forecasting were applied to a simplified version of the SRC: a method based on Gaussian process regression, a Bayesian structural time series model, and a dynamic linear model. Although we had initially expected the BSTS to perform the best, we ultimately found that it's performance was underwhelming. While the VI methods in `tfp` were useable given our computational constraints, the forecasting test performance was abysmal. In contrast, the MCMC inference methods implemented in `bsts` produced good results, but were ultimately too slow and inefficient for rapid development, testing and assessment. We found that the GPR and DLM approaches worked quite well, even though these models are in some ways less specifically tailored to time-series forecasting than BSTS.

Both the GPR and DLM methods resulted in test-set skills that were comparable to the best results from [2] (and even better, in some cases). We would, however, caution the reader to not interpret this as a claim that GPR and DLM are superior methods to the models used by [2] and the other competitors in the SRC. For one, we have selected a very small subset of the data to study - only training and testing on a single point in a grid of 514 points. Thus, it is possible that our single-point performance is not representative of how these models would transfer to the whole dataset (although we expect, on average, that the performance would be the same). Related to this point is that the skills calculation differs subtly between this project and in the SRC - as the method in Section 1.1, the SRC calculates anomaly vectors over all grid points, while we calculate anomaly vectors over multiple timesteps. We expect that on average these calculations amount to the same expected average skill, however it is likely that this discrepancy is non-trivial and should be considered when comparing our results to those of [2].

While our project focuses on a subsampled dataset and only made temperature predictions for a single geographic location, there is no fundamental reason that these methods could not be applied to the full-scale problem. This would then open the door to a direct comparison with [2]. In order to make a full comparison, the methods in [2] should be faithfully recreated. We close with a remark that the Bayesian methods explored in this project offer an important and substantial advantage over the more frequentist approach in [2]. Specifically, all the methods studied naturally result in a quantification of uncertainty - this could be vitally important if these forecasting models are to be used in industry/policy decisions. Given a very small posterior predictive variance, an analyst may be more confident in making an important decision based on the model, while a large variance may indicate to the analyst that the model is not to be trusted for that prediction.

References

[1] Christopher J White, Henrik Carlsen, Andrew W Robertson, Richard JT Klein, Jeffrey K Lazo, Arun Kumar, Frederic Vitart, Erin Coughlan de Perez, Andrea J Ray, Virginia Murray, et al. Potential applications of subseasonal-to-seasonal (s2s) predictions. *Meteorological applications*, 24(3):315–325, 2017.

[2] Jessica Hwang, Paulo Orenstein, Judah Cohen, Karl Pfeiffer, and Lester Mackey. Improving subseasonal forecasting in the western us with machine learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2325–2335, 2019.

[3] Andrew W Robertson, Arun Kumar, Malaquias Peña, and Frederic Vitart. Improving and promoting subseasonal to seasonal prediction. *Bulletin of the American Meteorological Society*, 96(3):ES49–ES53, 2015.

[4] Frédéric Vitart, Andrew W Robertson, and David LT Anderson. Subseasonal to seasonal prediction project: Bridging the gap between weather and climate. *Bulletin of the World Meteorological Organization*, 61(2):23, 2012.

[5] Sub-seasonal climate forecast rodeo. `https://www.challenge.gov/challenge/sub-seasonal-climate-forecast-rodeo/`.

[6] Weizhong Yan, Hai Qiu, and Ya Xue. Gaussian process for long-term time-series forecasting. In *2009 International Joint Conference on Neural Networks*, pages 3420–3427. IEEE, 2009.

[7] Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29(5-6):594–621, 2010.

[8] Steven L Scott and Hal R Varian. Predicting the present with bayesian structural time series. *International Journal of Mathematical Modelling and Numerical Optimisation*, 5(1-2):4–23, 2014.

[9] Marko Laine. Introduction to dynamic linear models for time series analysis. In *Geodetic Time Series Analysis in Earth Sciences*, pages 139–156. Springer, 2020.

[10] Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[11] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[12] Andrew C. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press, 1990.

[13] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*. Oxford university press, 2012.

[14] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[15] Joshua V Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A Saurous. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, 2017.

[16] Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474, 2017.

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Steven Scott. bsts: Bayesian structural time series. `https://cran.r-project.org/web/packages/bsts/index.html`.

[19] Giovanni Petris. dlm: an r package for bayesian analysis of dynamic linear models. *University of Arkansas*, 2009.

[20] Mike West and Jeff Harrison. The dynamic linear model. *Bayesian Forecasting and Dynamic Models*, pages 97–142, 1997.

[21] Pydlm: A python library for bayesian time series modeling. `https://github.com/wwrechard/pydlm`.

[22] Jessica Hwang, Paulo Orenstein, Judah Cohen, and Lester Mackey. The SubseasonalRodeo Dataset, 2018.

[23] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *arXiv preprint arXiv:1206.2944*, 2012.

[24] Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014–.