Bayesian Hierarchical Modeling for Equitable Marathon Qualifying Standards

Project Category: Athletics and Sensing Devices

Name: Spencer Siegel
SUNet ID: siegels
Department of Computer Science
Stanford University
siegels@stanford.edu

1 Introduction

The Boston Marathon qualifying standards have received quite a bit of scrutiny over the years. Many members of the running community question the fairness of the qualifying times for certain male and female age groups. Others have suggested that certain races which are eligible to qualify a runner are much easier than others. The current Boston qualifying standard for males and females are split into 11 age groups with standards that increase at an increasing rate.

As a senior at UNC Chapel Hill in 2020 (with guidance from Dr. Richard Smith), I began a study which utilized a linear mixed-effects model to produce aging curves for male and female marathon runners [1]. This research modeled each marathon location separately which inhibited the reliability and generalizability of the results. Furthermore, it is unable to control for the location of the race in evaluating the strength of a runner's performance.

This study extends my prior research by fitting a Bayesian Hierarchical Model (BHM) to predict marathon performance from 11 of the most popular marathons in the United States. The input to the algorithm is a tabular dataset of runner performances in these races from 2000-2019. The dataset includes gender, age, marathon location, and marathon year. I fit a separate model for male and females, treating a cubic spline on age as a fixed effect to account for the non-linear relationship between age and finish time. I model the individual runner ability, the marathon location, and the marathon location-year as random effects. With this specification, I fit a BHM using the numpyro package to output a distribution of predicted marathon times (in minutes) for each runner performance. For baseline comparisons, I also fit a ridge regression and CatBoost regressor on this same dataset.

2 Related work

Prior research in running performance has been focused on recommendations for training plans. Feely et al. detail a recommender system to provide training plans by using case-based reasoning (CBR) [1]. Although this work differs in purpose from my study, the first step in building this recommender system is to make real time predictions for a runner's marathon performance based on their current training schedule. While they added some complexity, the predictions were essentially derived from a nearest neighbors approach, learning from similar training practices. Knechtle et al. studied training regiments for half-marathons with a focus on physiological factors [2]. Particularly, they provide basic equations from hypothesis testing and regression analysis which predicts half-marathon performance from speed during training and body fat percentage. This is a quite useful and thought provoking study as it shows the expected benefits of losing body fat on running performance.

The most similar analysis to my study is the research by Waldovgel et al. which studied ultra-marathon (50 or 100 mile races) performance [3]. The study also used a mixed-effects modeling approach to fit a random effect on the individual runner ability. This study models males and females together and creates sex-country aging curves which indicate that the difference between male and female runners becomes more narrow as age increases. The country effect is particularly interesting because it helps account for cultural impact on training for males and females. Leyk et al. also found a related result in their simple regression analysis which shows that older athletes do not decline rapidly in half-marathon and marathon [4]. This study modeled performance with age groups and showed no significant performance drop off until the age of 50, with only small drop offs thereafter. These findings supported previous research from Young et al. which concluded that plasticity levels in older athletes decline moderately [5]. Through polynomial regression analysis in a longitudinal study, they show that the rate of age-related performance decline is more linear than quadratic for 45-69 year old runners.

My research is novel in that it is focused on marathon qualification standards and incorporates race and race-condition effects. Additionally, I utilize a Bayesian framework which allows for sampling the full posterior. Munkby compared probabilistic programming frameworks (PPFs) and showed significant performance benefits from numpyro over other PPFs like Stan and TensorFlowProbability [6].

3 Dataset and Features

Researchers at Colorado School of Mines under the supervision of Dr. Dorit Hammerling have scraped marathon data from 2000-2019 for 11 of the major United States marathons (Boston, Chicago, California International, Grandmas, Honolulu, Houston, Los Angeles, Marine Corps, New York, Philadelphia, Twin Cities). However, I also had to perform extensive data work due to the absence of unique runner IDs. The task to re-identify a runner as the same person from two different race observations is not trivial.

To build my large marathon dataset in which I re-identify runners from different race locations and years, I began by creating unique IDs. The data includes over 3.5 million marathon observations which span the years 2000-2019 and 11 locations across the United States. The fields in the data include age (year as a discrete variable), name, gender, race location, and race year. I was forced to use age, name, and gender to match runners from different races and years. My approach for this task was that I would rather throw away data in questionable cases than re-identify runners incorrectly. The re-identification of runners is critical in order to model individual runner ability.

My first step in solving this task was to create individual marathon id numbers (i.e. Each id corresponds to a particular runner ran in all of the years of a given race location such as the Chicago Marathon). It is much easier to recognize runners from year to year in the same race location because they can only run one race in a given year and their age generally should increase by one each year. I indexed on name, gender, and age to create a runner-marathon id. Next, I wanted to re-identify runners from different races so I created a global runner id by attempting to match runners with these three keys. However, I was flexible on the age component and allowed matches for runners of the same gender and name that were one year older or younger in the same year (since your age can increase in the same year) depending on the time of the race. I also took a few precautionary measures. I removed global runner ids that had 3 ages in the same year (clearly a mistake in identification) and I removed runners who had races with times greater than 100 minutes apart. This was chosen arbitrarily but my goal was to reduce the number of incorrectly identified runners. Ultimately, this left me with a dataset with 2,960,345 total observations, of which 60.4% were male and 39.6% female. About 38.8% of runners in the dataset participated in multiple races. As shown below, most runners in the data were between 30 and 50 years-old with very few older runners.

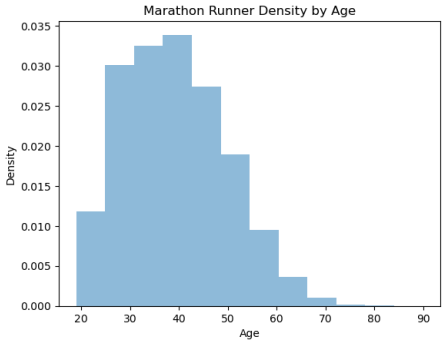


Figure 1: Marathon Runner Age Density Plot

4 Methods

4.1 Ridge

My first modeling task with the processed dataset was to create a baseline linear model that ignores the individual runner. Namely, I modeled the continuous response of marathon time measured in minutes as a function of age and marathon location-year (ideally this will serve as a proxy for weather and other circumstances particular to race day). I fit separate models for males and females since the focus of this study is to generate aging curves aging curves for each sex to advise marathon boards rather than compare times between sexes. For the aging aspect, I used a cubic spline on age with 5 degrees of freedom (creates 6 equally spaced knots) to represent ages from 18-85 (the ages of interest in this study). This is an interpolation method which uses third degree polynomials to guarantee smoothness where they join at the 6 equally spaced knots. The functions have continuous first and second derivatives. This can account for the non-linear relationship between age and marathon time. I used a ridge regression which places a L2 penalty on the model coefficients as a regularization technique to penalize large coefficients. My model was specified as follows:

$$T_{ijk} = \theta_0 + \sum_{l=1}^{5} S(a_{ijk}; 5)_l \theta_l + M_{jk} + \lambda ||\theta||_2^2$$

Where T_{ijk} represents the time measured in minutes for the particular race. θ_0 represents the intercept. $S(a_{ij}; 5)$ represents a cubic spline with 5 degrees of freedom on the age of runner i in marathon j in year k. θ_l for l=1,..., 5

represents the regression coefficients on the aging spline. M_{jk} is the coefficient for the kth year of the jth marathon location. λ is the L2 regularization hyperparameter.

4.2 CatBoost

Next, I fit a CatBoost regression with the same features as above and an additional categorical feature for the global runner id. I decided to utilize this model because it handles categorical features very well and allowed me to avoid one-hot-encoding all of the individual runners. CatBoost is a gradient boosting decision tree algorithm. I used squared error as the lost function. The algorithm sequentially constructs decision trees as the "weak" learner. For each tree, it calculates the gradient of the loss function with respect to the model parameters and moves in the direction of the negative gradient for the next tree (additive model). The ensemble of these "weak" learners creates a "strong" learner which is used in prediction. CatBoost handles categorical features (like global runner id in this use case) through target encoding. It first permutes the training data in a random order. Then it converts the categorical classes to numeric values through quantization. The quantization method used in this application is called "GreedyLogSum" in which the following equation is maximized within each category class k (each runner id): $\sum_{i=1}^{n} log(weight)$ where n is the number of distinct classes in the training dataset (number of distinct runners) and weight is the number of appearances of class k in the training dataset (number of observations for a particular runner k). For each training example, the algorithm only uses previous examples in the permutation to calculate this value (the ordered target statistic). These values are then encoded to use in the algorithm for a particular permutation of the data. Several permutations of the data are defined and a random one is chosen for each iteration of the learning algorithm.

4.3 Bayesian Hierarchical Model

Lastly, I implemented a BHM using the numpyro Python package. This is a very flexible linear modeling technique that allows for both fixed and random effects to be used in a Bayesian setting. Fixed effects refer to traditional regression coefficients. For this model specification, I used a fixed effect for the aging spline. Random effects are often used on categorical variables in which one is trying to control and where the categories in the training data represent a subset of the total population. Each level of the categorical variable is treated as a draw from a global distribution. Furthermore, using a random effect is a method to account for non-independent data. In a traditional regression setting, we assume that observations are independent, but in reality, observations may be dependent if they share certain characteristics (such as sharing the same level of some random effect).

This is a perfect application for individual runners where I want to represent individual runner skill, but also want the model to generalize to unseen runners. Clearly, races run by the same runner are highly correlated. Through this random effect structure, I can estimate separate coefficients for each runner where coefficients are drawn from a normal distribution. Additionally, I utilize a nested random effect structure to represent the marathon location and marathon location-year effect. Ideally, the parameter representing the marathon location would represent the difficulty of the particular course and the year effect would represent the conditions (perhaps weather) for a particular year. A nested random effect structure focuses on the difference between groups and the difference among groups. I estimate the marathon course with a random effect intercept but also include a random effect on the year of the race that is nested within marathon course. Once again, this random effect structure is intuitive and allows for estimation for new race locations and years that are not found in my training set. I have used a probabilistic programming language (numpyro) to model in a Bayesian manner, thereby estimating the full posterior of the modeling parameters. This allows for sampling of the modeling parameters for prediction. To estimate the full posterior distribution of the parameters, I used stochastic variational inference with 10,000 steps. Hoffman et al. proposed this optimization technique which iterates between subsampling the data and adjusting the hidden structure based on the subsample [7]. The adjustment to the parameters is made in the direction that minimizes the Kullback-Leibler (KL) divergence between the variational and posterior distributions. I was not able to tune the number of steps that I used for the algorithm due to computational complexity. The specification of the model that I used is given by the following equation:

$$T_{ijk} \sim TruncatedNormal(\theta_0 + \sum_{l=1}^{5} S(a_{ijk}; 5)_l \theta_l + \epsilon_i + \epsilon_j + \epsilon_{k|j}, \sigma^2)$$

Where T_{ijk}, θ_0, $S(a_{ij}; 5)$, θ_l are identical to the ridge setting. I utilized a truncated-normal for the response since times cannot fall below zero minutes. $\epsilon_r \sim N(0, \sigma_r^2)$ is the random effect intercept on runner i. $\epsilon_j \sim N(0, \sigma_m^2)$ is the random effect intercept on marathon j. $\epsilon_{k|j} \sim N(0, \sigma_y^2)$ is the random effect on year k nested within marathon j.

5 Experiments

I began by randomly splitting the marathon dataset into a train (80%), validation (10%), and test set (10%). These same training, validation set (when applicable), and test sets were used to evaluate all three modeling approaches for

comparison sake. For evaluation on the test sets, I used regression loss metrics such as r-squared, mean squared error (MSE), and mean absolute error (MAE) with a focus on how well each learning algorithm predicts on each age groups. I plotted gender-aging curves and evaluated the new recommended Boston Marathon qualifying standards under each model.

5.1 Ridge and CatBoost

For the ridge regression, I implemented 10-fold cross validation to tune lambda (the regularization hyperparameter). For the gender-aging curves, I isolated age and predicted marathon time using aging coefficients for each age (from 18-90) and intercept coefficients.

For CatBoost, I utilized a Bayesian optimization algorithm for hyperparamter tuning with the python package optuna. Essentially it searches over a user-determined hyperparameter space by trial and error. uses the past history to intelligently try a new set of hyperparameters until it reaches a specified number of iterations. For CatBoost, I performed tuning for many important hyperparamters: learning rate, maximum depth of trees, l2 leaf regularization, and the minimum number of training samples in a leaf. I tuned these parameters by optimizing for mean absolute error performance on the validation set. For gender-aging curves, I predicted marathon time using a new fictitious marathon-year and runner id (unknown categories) to isolate the age variable to use for the aging curve.

5.2 Bayesian Hierarchical Model

I was not able to perform any cross validation for this model to tune the standard deviations due to computational complexity. I passed in priors for the standard deviations based on intuition: 100.0 for σ_r, 5.0 for σ_m, 10.0 for σ_y. These values essentially serve as regularizers and I was unable to spend time tuning theses on the validation set but this is an obvious area of improvement. Lastly, as in ridge, I isolated the intercept and aging variables to predict marathon times for the gender aging curves. To obtain approximations for the coefficients I generated 1,500 samples of the posterior distribution and obtained the mean for each parameter. More analysis could be done on the distribution of these parameters but for now I obtain point estimates of the parameters from the posterior means.

6 Results/Discussion

As anticipated, the error metrics on the test set (MSE, MAE, r-squared) of the ridge regression performed much worse than that of catboost and numpyro. However, it is very encouraging to see the prediction metrics across the board perform much better with the BHM than CatBoost for both male and female runners. Furthermore, BHM outperforms the other models significantly across all age groups. I have shown the MAE differences in Figure 2 for males, but the same pattern holds true for female predictions. This is especially important considering the purpose of this study is to generate accurate aging curves to consult marathons on appropriate qualifying standards.

	Ridge	CatBoost	BHM
Male RMSE	45.648	38.816	36.774
Male MAE	36.745	30.283	27.613
Male R-Sq	0.169	0.399	0.4609
Female RMSE	42.752	36.532	34.786
Female MAE	34.584	28.763	26.636
Female R-Sq	0.169	0.393	0.450

Table 1: Test Set Error Metric Comparison

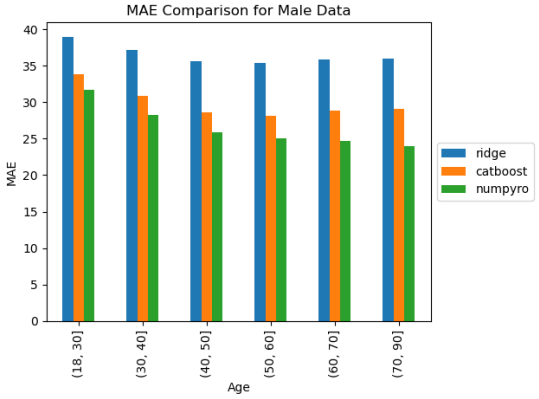
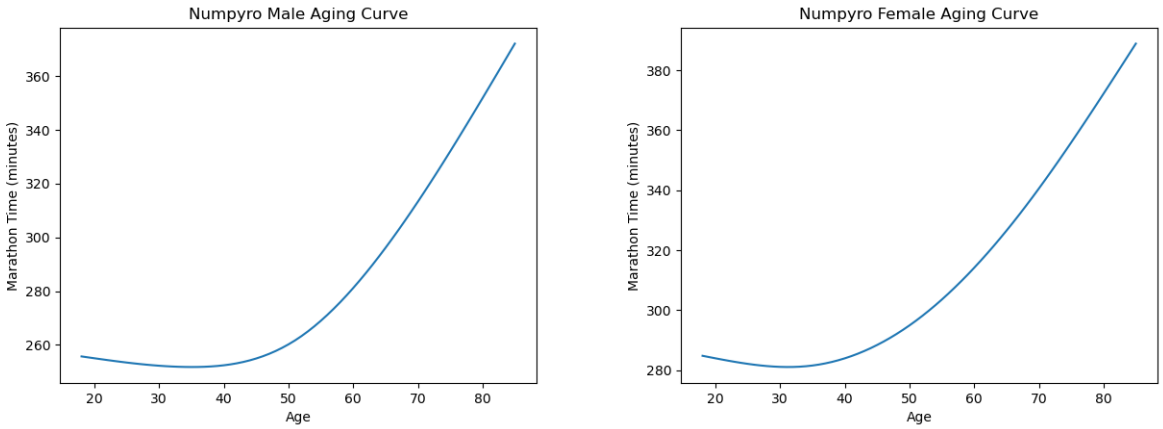


Figure 2: MAE comparison by Age Group

Inspecting many of the model coefficients from the BHM also provides insights on potential concerns with the current Boston Marathon qualifying standards which ignore race location and year. For instance, the marathon random effect coefficients suggest that an average male and female runner in the California International Marathon will run 10.18 and 13.75 minutes faster respectively than on an average course. This course is known to be downhill, so this effect is intuitive. The model also shows that the Honolulu marathon is a difficult course (likely due to weather) and average male and female runners see an increase of 19.18 and 17.78 minutes in their finish time. Additionally, through the

nested marathon-year coefficients, the model highlights certain years where marathon conditions were more difficult such as the Boston Marathon in 2004. This race was known for extremely hot temperatures [8]. These results have significant implications considering the thin margins in qualifying for the Boston Marathon. The current standards do not take the marathon course or year run (conditions) into account which appears problematic.

I have plotted gender aging curves from the BHM. These results show that performance begins to decline around the age of 40 and declines at an increasing rate with age. One potential weakness of this model is the lack of data for older runners. Although the test set performance on older runners remained consistent with younger groups, there is a much smaller sample size. Differences in training set and test set performance appear significant, but this is due to the inclusion of the individual runner effect. However, it appears that the BHM is not overfitted due to the logical shape of the aging curve. CatBoost, on the other hand, appeared to overfit by suggested that older runners improve. This is likely due to survivorship bias where slower runners are unable to continue to race as they age. As seen below, BHM does not appear to suffer from this bias.



Figures 3-4: Male and Female Gender Aging Curves

Age Group	18-34	35-39	40-44	45-49	50-54	55-59	60-64	65-69	70-74	75-79	80+
Male BQ	3:00	3:05	3:10	3:20	3:25	3:35	3:50	4:05	4:20	4:35	4:50
Male BHM	3:00	2:59	3:00	3:03	3:09	3:20	3:37	3:49	4:07	4:27	4:49
Female BQ	3:30	3:35	3:40	3:50	3:55	4:05	4:20	4:35	4:50	5:05	5:20
Female BHM	3:30	3:30	3:33	3:38	3:45	3:55	4:06	4:19	4:34	4:50	5:08

Table 2: Boston Qualifying (BQ) times compared to BHM Recommended Qualifying Times (Times in Hr:Min)

Lastly, I computed recommended qualifying times based on the aging curves. I made a key assumption that the shape of the aging curve holds true for runners of various skill levels. I tested this assumption by fitting my model on only the top 50% of runners and found similar aging curves compared to using the full dataset. To compare to the current Boston Marathon standards, I set the first age group of 18-34 years old equal to the current standards for males and females. Then, for each age group, I used the aging curve to define how this runner would perform as he/she ages by computing the average difference from the previous age group. While this makes a few assumptions, it is clear from the results of Table 2 that the current standards seem to be too harsh on younger runners relative to middle-aged runners. The BHM results are stricter than BQ standards for all age groups (suggesting that the current 18-34 standards are too harsh) but the BHM suggests greater decline for ages 69-80+. For both males and females, the BHM results suggest that the qualifying times should not be increasing much at all for runners before the age of 50 contrary to current standards.

7 Conclusion/Future Work

A Bayesian Hierarchical Modeling approach to predicting marathon performance has shown to be quite promising. The performance gap between CatBoost and BHM indicates that the linearity assumption between the features and finish time is appropriate and that the hierarchical structure is effective. Furthermore, the results suggest that race location and year should be considered when evaluating one's performances. Also, the BHM results suggest that the current standards are too strict on young runners relative to middle-aged runners in particular. Having more data for older runners and spending more time tuning and utilizing the full posterior could further strengthen the findings of the study. However, these results suggest many tweaks to the current marathon qualifying standards to address fairness.

8 Contributions

This project was completed by Spencer Siegel.

References

[1] Ciara Feely, Brian Caulfield, Aonghus Lawlor, and Barry Smyth. Providing explainable race-time predictions and training plan recommendations to marathon runners. In *Fourteenth ACM Conference on Recommender Systems*, pages 539–544, 2020.

[2] B Knechtle, U Barandun, P Knechtle, MA Zingg, T Rosemann, and CA Rüst. Prediction of half-marathon race time in recreational female and male runners. springerplus. 2014; 3: 248.

[3] Karin J Waldvogel, Pantelis T Nikolaidis, Stefania Di Gangi, Thomas Rosemann, and Beat Knechtle. Women reduce the performance difference to men with increasing age in ultra-marathon running. *International journal of environmental research and public health*, 16(13):2377, 2019.

[4] D Leyk, O Erley, D Ridder, M Leurs, T Rüther, M Wunderlich, A Sievert, K Baum, and D Essfeld. Age-related changes in marathon and half-marathon performances. *International journal of sports medicine*, 28(06):513–517, 2007.

[5] Bradley W Young and Janet L Starkes. Career-span analyses of track performance: longitudinal data present a more optimistic view of age-related performance decline. *Experimental Aging Research*, 31(1):69–90, 2005.

[6] Carl Munkby. Evaluation of probabilistic programming frameworks, 2022.

[7] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

[8] Abraham J Miller-Rushing, Richard B Primack, Nathan Phillips, and Robert K Kaufmann. Effects of warming temperatures on winning times in the boston marathon. 2012.

[9] G David Garson. *Hierarchical linear modeling: Guide and applications*. Sage, 2013.

[10] Python Core Team. *Python: A dynamic, open source programming language*. Python Software Foundation, 2019.

[11] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[12] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[13] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[14] Du Phan, Neeraj Pradhan, and Martin Jankowiak. Composable effects for flexible and accelerated probabilistic programming in numpyro. *arXiv preprint arXiv:1912.11554*, 2019.

[15] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul A. Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep universal probabilistic programming. *J. Mach. Learn. Res.*, 20:28:1–28:6, 2019.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 2020.

[18] Liudmila Ostroumova Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *NeurIPS*, pages 6639–6649, 2018.

[19] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[20] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.