Massively Parallel Approximate Gaussian Process Regression

Jarad Niemi

Iowa State University

May 25, 2016

with Bobby Gramacy and Robin Weiss, University of Chicago

Nonparametric regression

Given a set of input-output pairs $(x_1, y_1), \ldots, (x_N, y_N)$ with $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$, we assume

$$E[y|x] = f(x)$$

for some unknown function f.

We consider the scenario where

- $p \sim 10$
- $N \sim 10,000$

and our primary goal is to predict a set of outputs $\tilde{y}_1, \ldots, \tilde{y}_{\tilde{N}}$ based on their associated inputs $\tilde{x}_1, \ldots, \tilde{x}_{\tilde{N}}$.

Gaussian process regression

Gaussian Process regression provides a prior over functions $f : \mathbb{R}^p \rightarrow \mathbb{R}$ where

- any finite collection of outputs (y) are jointly Gaussian,
- defined by a mean function $\mu(x) = E[Y(x)] = 0$, and
- and covariance function

$$\begin{aligned} C(x, x') &= E\left\{[Y(x) - \mu(x)][Y(x') - \mu(x')]^\top\right\} \\ &= \tau^2 K_\theta(x, x'). \end{aligned}$$

We utilize the isotropic Gaussian correlation

$$K_\theta(x, x') = \exp(-||x - x'||^2/\theta)$$

where θ is referred to as the *lengthscale* parameter.

Gaussian Process regression estimation

Let $y = (y_1, \ldots, y_n)$ and $X = (x_1^\top, \ldots, x_n^\top)$, then

$$y \sim N(0, \tau^2 K_\theta) \quad \text{where} \quad K_\theta(i,j) = K_\theta(x_i, x_j).$$

If $p(\tau^2) \propto 1/\tau^2$, then

$$L(\theta) \propto p(y|\theta) = \int p(y|\theta, \tau^2) p(\tau^2) d\tau^2 = \frac{\Gamma[N/2]}{(2\pi)^{N/2} |K_\theta|^{1/2}} \times \left(\frac{\psi_{\theta,y}}{2} \right)^{-\frac{N}{2}}$$

where $\psi_{\theta,y} = y^\top K_\theta^{-1} y$. Analytic derivatives allows for easy maximization of this (marginal) likelihood, i.e.

$$\hat{\theta} = \text{argmax}_\theta p(y|\theta).$$

Gaussian Process regression prediction

The predictive distribution for $\tilde{y}$ conditional on $\hat{\theta}$ and $\tilde{x}$ is Student t with N degrees of freedom, mean

$$\hat{f}(\tilde{x}) = \mu(\tilde{x}|\hat{\theta}, y) = k_{\hat{\theta}}^{\top}(\tilde{x}) K_{\hat{\theta}}^{-1} y,$$

and scale

$$\sigma^2(\tilde{x}|\hat{\theta}, y) = \frac{\psi_{\hat{\theta},y}[K_{\hat{\theta}}(\tilde{x}, \tilde{x}) - k_{\hat{\theta}}^{\top}(\tilde{x}) K_{\hat{\theta}}^{-1} k_{\hat{\theta}}(\tilde{x})]}{N}$$

where $k_{\hat{\theta}}^{\top}(x)$ is the N-vector whose ith component is $K_{\hat{\theta}}(\tilde{x}, x_i)$.

Define $V(\tilde{x}) \equiv Var[\tilde{y}|\hat{\theta}, y] = \sigma^2(\tilde{x}|\hat{\theta}, y) \times N/(N-2)$.

Issues

- Estimation and prediction are computationally intractible when N is large due to the $O(N^3)$ matrix operations, i.e. $|K_\theta|$ and K_θ^{-1}
- Gaussian process regression model enforces stationarity, i.e. the same covariance properties throughout the domain.

Nearest neighbors

When trying to predict $\tilde{y}$ for input $\tilde{x}$, rather than using y and X consider using a subset of the original data. If we choose the n nearest neighbors, let

$$D_n(\tilde{x}) = \{(x_i, y_i) : K_\theta(\tilde{x}, x_i) \leq \delta_{(n)}\}$$

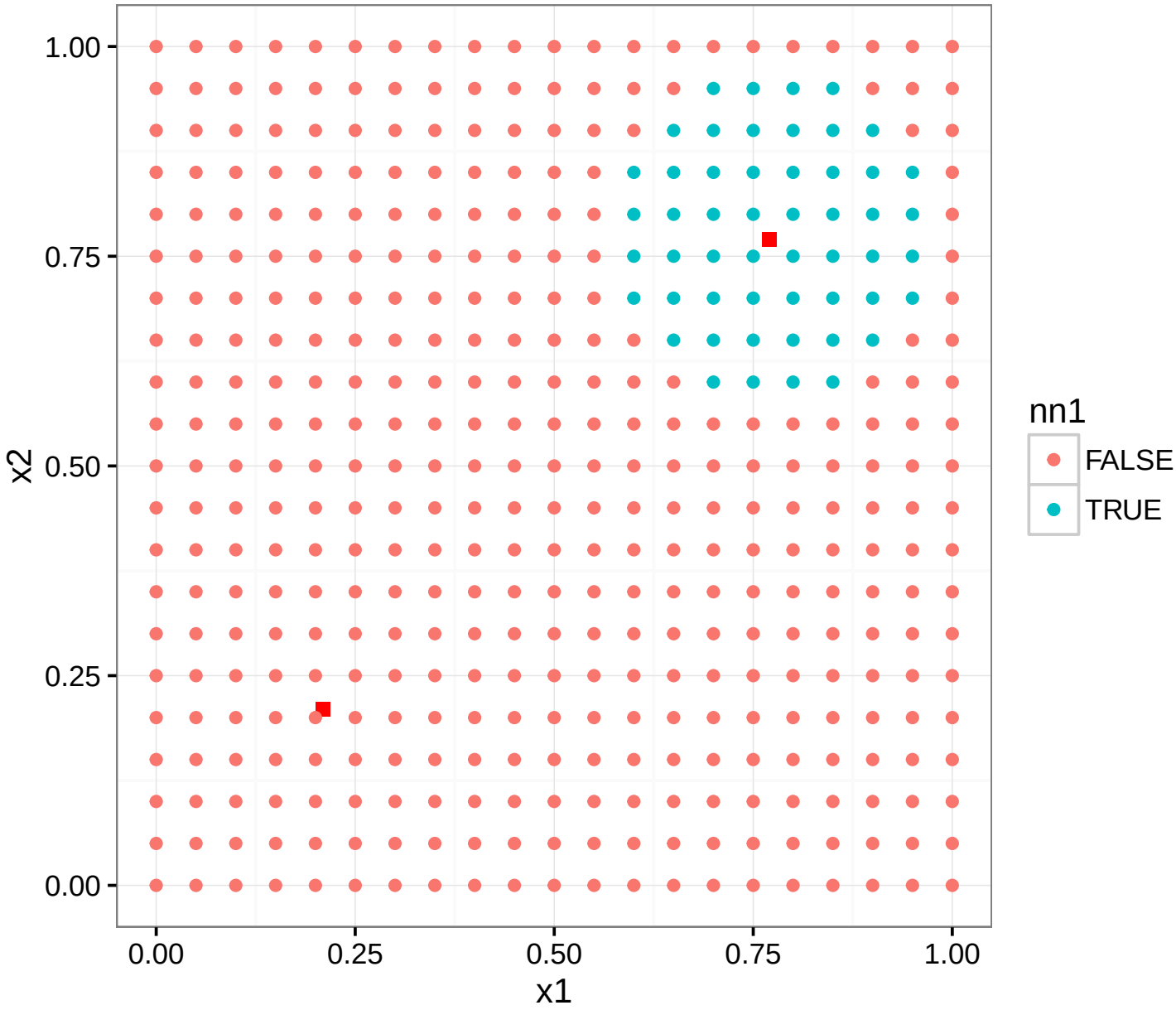
where $\delta_{(n)}$ is the nth order statistic for the distances between $\tilde{x}$ and each of the other locations x_i. For location $\tilde{x}$, base estimation and prediction on $D_n(\tilde{x})$ rather than $D_N(\tilde{x})$.

Both problems are solved since

- matrix operations are now order $O(n^3)$ and
- the model is only locally stationary.

In addition,

- $E[\tilde{y}|\tilde{x}, D_n(\tilde{x}), \hat{\theta}_{D_n(\tilde{x})}] \to E[\tilde{y}|\tilde{x}, D_N(\tilde{x}), \hat{\theta}_{D_N(\tilde{x})}]$ as $n \to N$
- $V[\tilde{y}|\tilde{x}, D_n(\tilde{x}), \hat{\theta}_{D_n(\tilde{x})}] > V[\tilde{y}|\tilde{x}, D_N(\tilde{x}), \hat{\theta}_{D_N(\tilde{x})}]$


1.00
0.75
0.50
0.25
0.00
x2
0.00
0.25
0.50
0.75
1.00
x1
nn1
FALSE
TRUE

Sequential design

- The nearest neighbor approach is sub-optimal collection of n points for prediction at input $\tilde{x}$.
- The optimal solution is computationally intractible due to the combinatorics involved, i.e. N choose n.

We can do better by solving a sequence of easier decision problems, i.e.

1. Choose an initial design n_0, call it $D_{n_0}(\tilde{x})$.
2. For $j = n_0, \ldots, n$,
   a. estimate θ based on $D_j(\tilde{x})$,
   b. given $D_j(\tilde{x})$, choose x_{j+1} according to some criterion and
   c. augment the design, $D_{j+1}(\tilde{x}) = D_j(\tilde{x}) \cup (x_{j+1}, y_{j+1})$.

Criterion

- Minimize the empirical Bayes mean-square prediction error:

$$\begin{aligned} J(x_{j+1}, \tilde{x}) &= E\{(\tilde{y}(\tilde{x}) - E[\tilde{y}|\tilde{x}, D_{j+1}, \hat{\theta}_{D_{j+1}}])^2 | D_j\} \\ &\approx V_j[\tilde{y}(\tilde{x})|D_{j+1}; \hat{\theta}_j] + \left(\frac{\partial \mu_j(\tilde{x};\theta)}{\partial \theta}\Big|_{\theta=\hat{\theta}_j} \right)^2 / \mathcal{G}_{j+1}(\hat{\theta}_j). \end{aligned}$$

- Maximize reduction in predictive variance:

$$v_j(x_{j+1}, D_j, \tilde{x}) = V_j[\tilde{y}(\tilde{x})|D_j; \hat{\theta}_j] - V_{j+1}[\tilde{y}(\tilde{x})|D_{j+1}; \hat{\theta}_j]$$

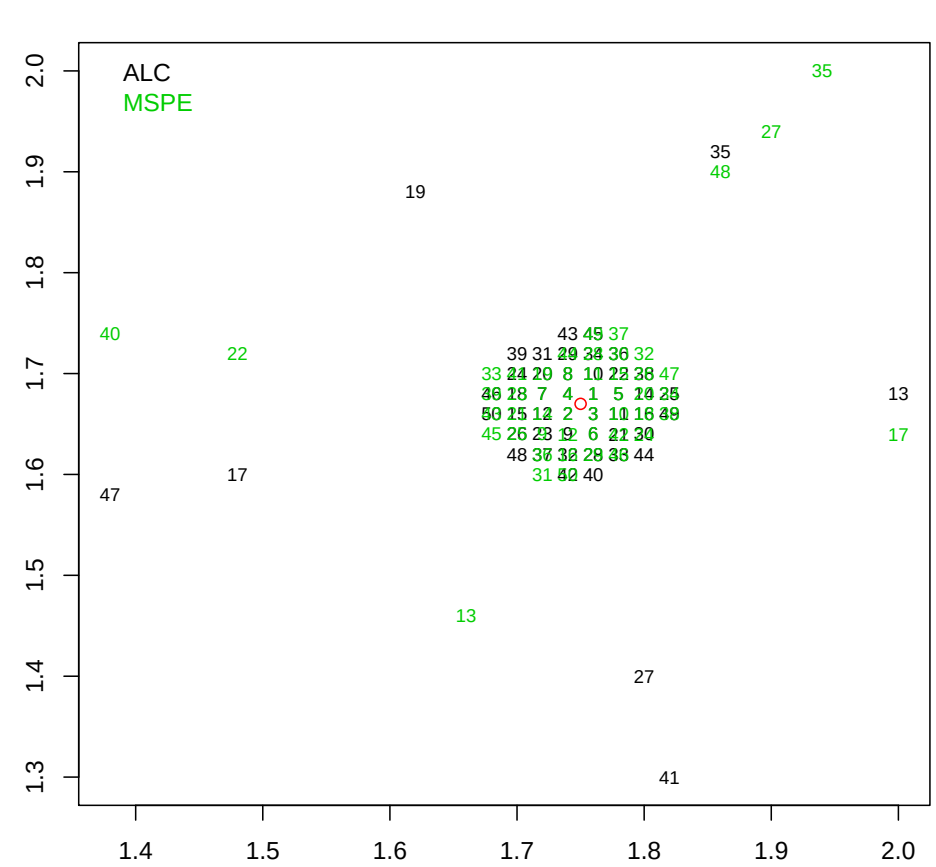
We call this active learning Cohn (ALC) (1996).

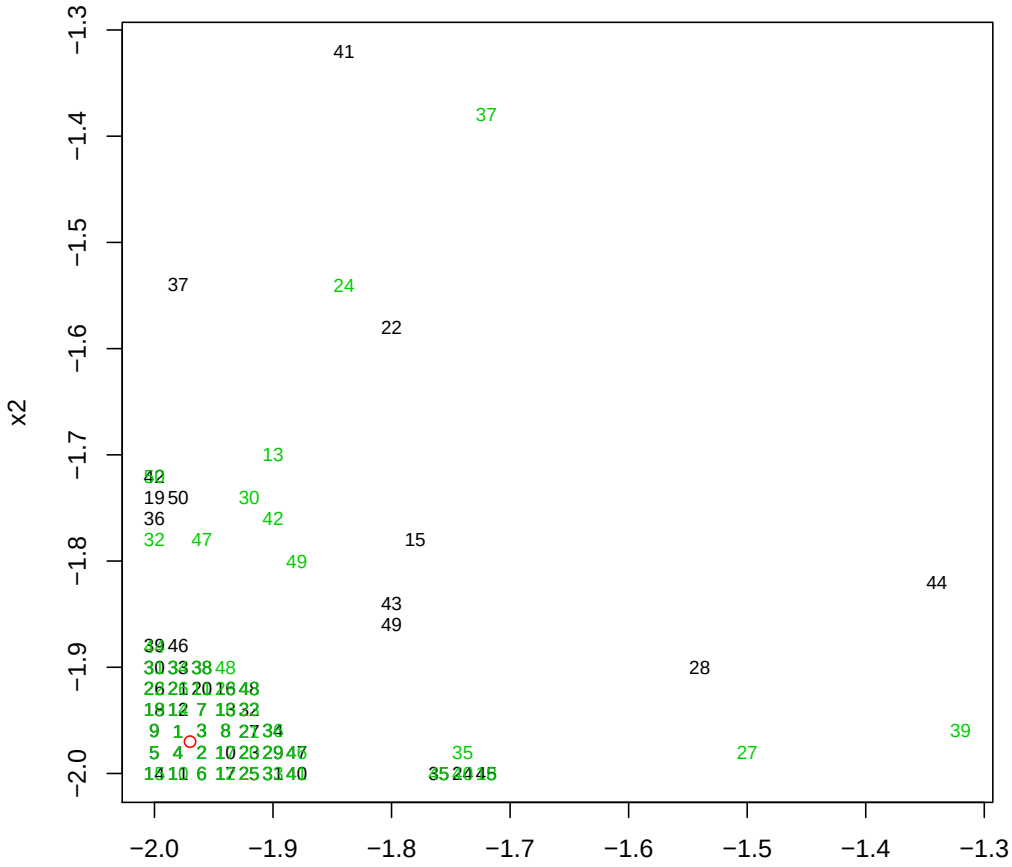
Points chosen by the criterion

$$f(x_1, x_2) = -w(x_1)w(x_2), \quad \text{where}$$

$$w(x) = \exp\left(-(x-1)^2\right) + \exp\left(-0.8(x+1)^2\right) - 0.05\sin\left(8(x+0.1)\right)$$

with X on a 201×201 ($= 40401$ point) regular grid in $[-2, 2]$.





Massively parallel

Problem: Predict a set of outputs $\tilde{y}_1, \ldots, \tilde{y}_{\tilde{N}}$ based on their associated inputs $\tilde{x}_1, \ldots, \tilde{x}_{\tilde{N}}$ where N and $\tilde{N}$ are both large.

Solution: For each location $\tilde{x}$, prediction from a Gaussian process regression model based on the design $D_n(\tilde{x})$ ($n << N$) where the inputs in the design are sequentially added based on the ALC criterion.

(Almost) embarrassingly parallel:

- Each location $\tilde{x}$ can be predicted independently.
- In each iteration of the sequential design, the potential locations can be evaluated independently and then compared.

Parallel computing

Parallelizing across prediction locations was relatively trivial with OpenMP:

```
#pragma omp parallel for private(i)
for(i=0; i<npred; i++) { ...
```

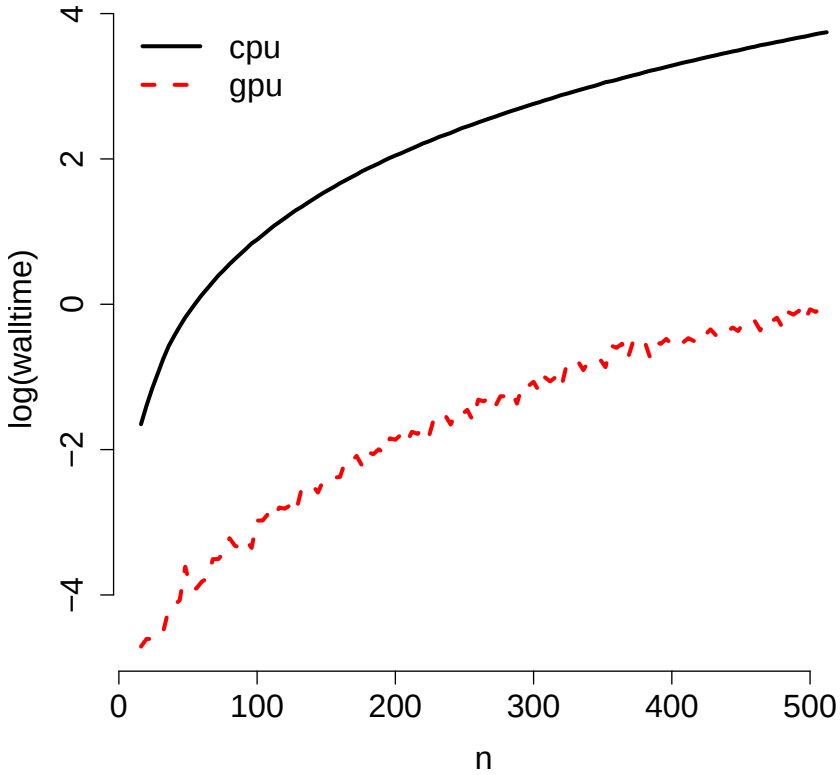
The ALC-based decision, i.e.

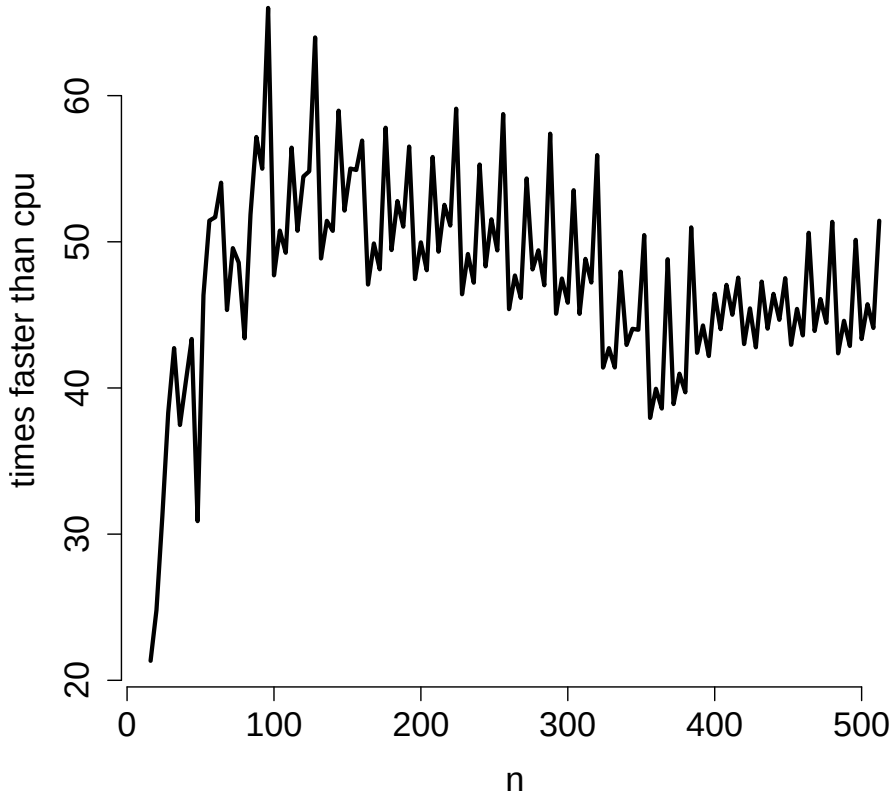
$$\text{argmin}_{x_{j+1} \in D_{N'} \setminus D_j} v_j(x_{j+1}, D_j, \tilde{x})$$

is off-loaded to a GPU with

- each x_{j+1} in a separate block with
- $O(j^2)$ linear algebra on j threads.

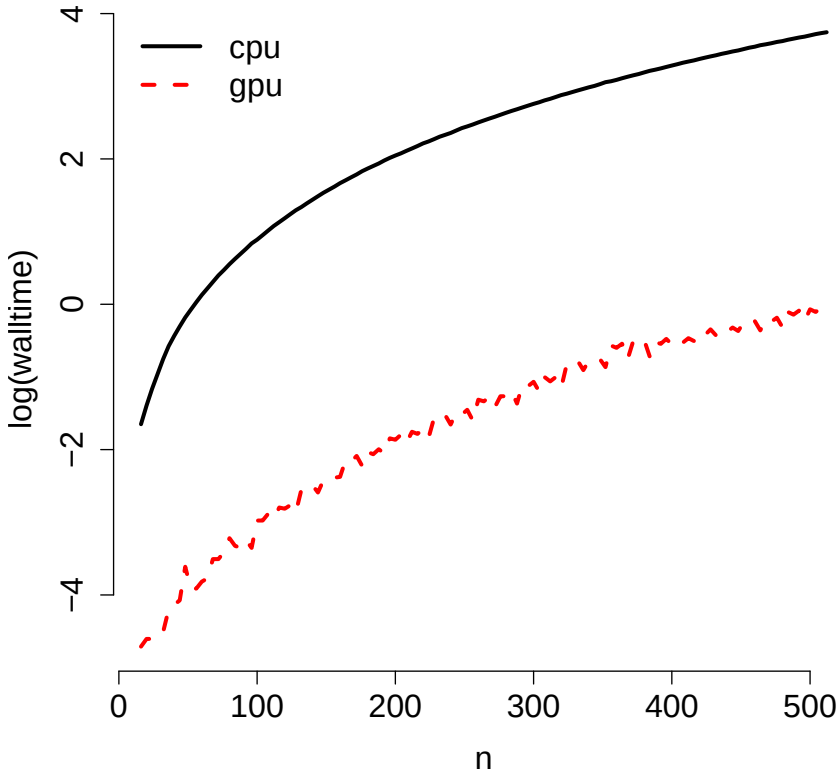
Computational comparisons

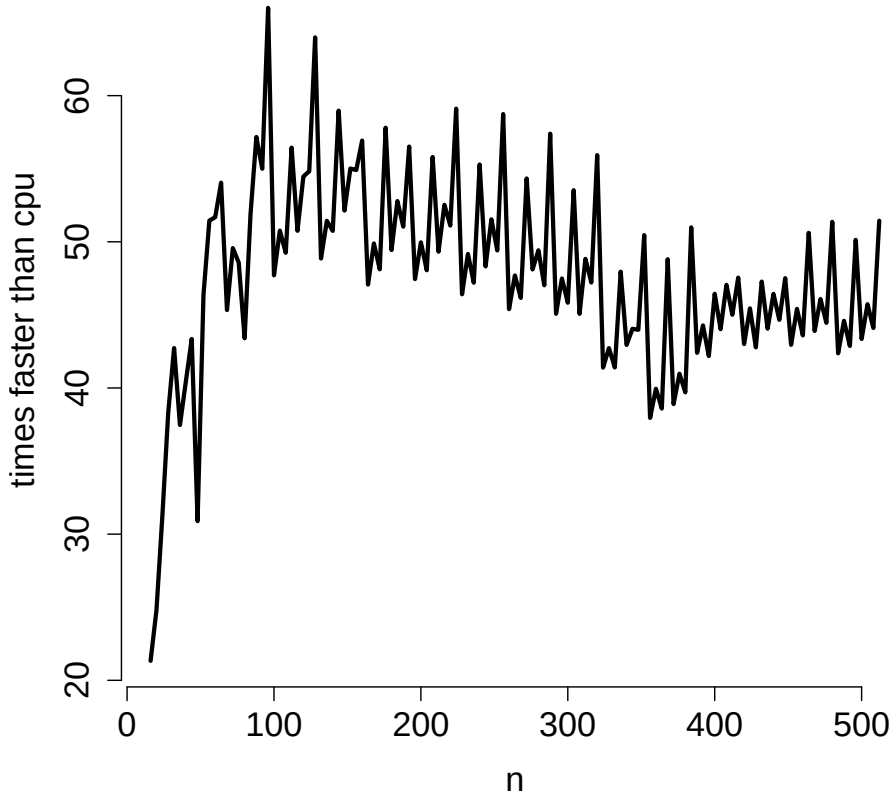




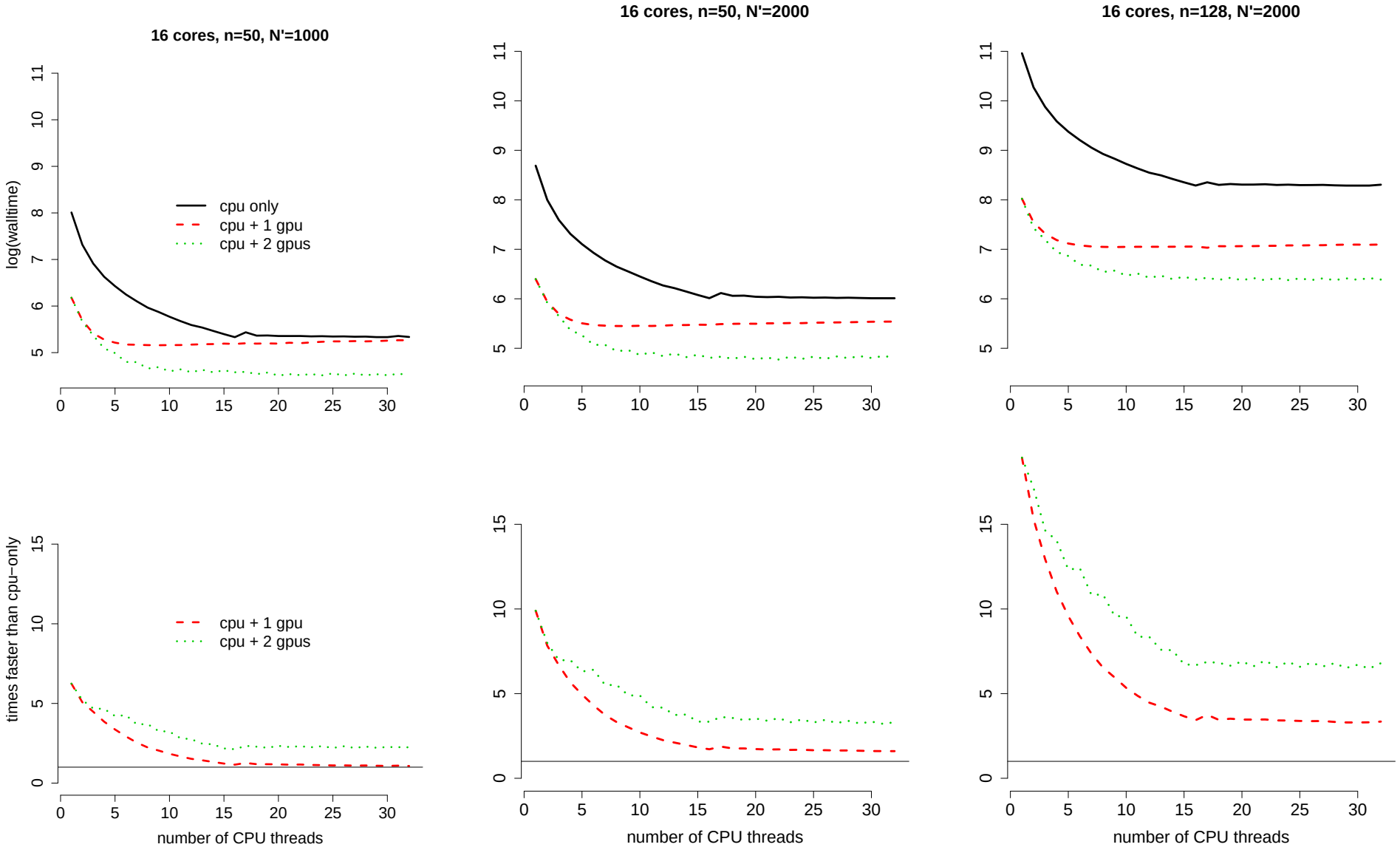
Comparing CPU-only and GPU-only wall-clock timings for the ALC calculation at each of $n_{\text{cand}} = 60000$ candidate locations for varying n, the size of the local design.

Computational comparisons





Comparing CPU-only and GPU-only wall-clock timings for the ALC calculation at each of $n_{\text{cand}} = 60000$ candidate locations for varying n, the size of the local design.



Comparing full global approximation times for a $\sim$ 40K-sized design, and a $\sim$ 10K predictive locations.

One hour super-computing budget

			96x CPU		5x 2 GPUs		$\frac{\text{CPU}}{5 \cdot \text{GPU}/96}$
N	n	N'	seconds	mse	seconds	mse	efficiency
1000	40	100	0.48	4.88	1.95	4.63	4.73
2000	42	150	0.66	3.67	2.96	3.93	4.26
4000	44	225	0.87	2.35	5.99	2.31	2.79
8000	46	338	1.82	1.73	13.09	1.74	2.66
16000	48	507	4.01	1.25	29.48	1.28	2.61
32000	50	760	10.02	1.01	67.08	1.00	2.87
64000	52	1140	28.17	0.78	164.27	0.76	3.29
128000	54	1710	84.00	0.60	443.70	0.60	3.63
256000	56	2565	261.90	0.46	1254.63	0.46	4.01
512000	58	3848	836.00	0.35	4015.12	0.36	4.00
1024000	60	5772	2789.81	0.26	13694.48	0.27	3.91

Resources

Gramacy, R. B., Niemi, J., & Weiss, R. M. (2014). Massively parallel approximate Gaussian process regression. SIAM/ASA Journal on Uncertainty Quantification, 2(1), 564-584.

This talk is available on my github site:
`https://github.com/jarad/SRC2016`.

Code to utilize the method is available in the `laGP` package on CRAN.

Thank you!