Efficient Gaussian Process Regression for Large Data Sets
A. Banerjee, D. Dunson, S. Tokdar (2011)

B. Cook & C. Hornbaker

Stat 225: Spatial Statistics

Harvard University

February 19, 2014

Overview

1. Introduction
2. Predictive Processes and Subset of Regressors
3. Random Projection Method
4. Properties of RP Method
5. Matrix Approximations & Projection Construction
6. Experimental Results
7. Code Example

Introduction

We have noisy observations $y_1, ..., y_n$ from the unknown function $f : \mathcal{X} \to \mathfrak{R}$ observed at locations $x_1, ..., x_n$, with some noise ϵ_i, so

$$y_i = f(x_i) + \epsilon_i, \text{ for } i = 1, ...n$$

We assume $f(\cdot)$ is a realization of a GP. To make predictions for other locations, we can use

$$g(x) = (k_{x,f})^T (K_{f,f})^{-1} \mathbf{y}_f$$

Where $K_{f,f} = cov\{f(X)\}$

Problem: This can be computationally expensive for large data sets...

- Matrix inversions in prediction operations are $O(n^3)$
- Data storage and processing leads to computational bottlenecks
- Numerical stability degrades as n increases

Introduction

Solution: Approximation techniques!

Several methods proposed:

- Predictive processes (Spatial statistics)
- Subset of regressors (Machine learning)

A. Banerjee, et. al.'s proposition:

- Random projection of all data points onto lower-dimensional subspace (Motivated by compressive sensing)

Predictive Processes and Subset of Regressors

Predictive Process: **"Knots" Method**

1. Select a set of "knots" in $\mathcal{X}$: $X^* = \{x_1^*, ..., x_m^*\}$
2. Evaluate $f(x)$ at the knots: $f^* = f(X^*) = \{f(x_1^*), ..., f(x_m^*)\}$
3. Predict by $g(x) = E\{f(x)|f^*\}$

Subset of regressors obtained via optimal approximation to $K_{f,f}$ with

$$Q_{f,f} = K_{f,*}(K_{*,*})^{-1}K_{*,f}$$

$g_{SOR}(\cdot)$ drawn from a GP with degenerate covariance kernel

$$q_{SOR}(x, z) = (k_{x,*})^T(K_{*,*}^T)^{-1}k_{*,z},$$

Considerations:

- Choice and spacing of knots can have substantial impact on results
- Methods allowing uncertain numbers and spacing of knots increase computational burden and negate savings of a low-rank method

Random Projection Method

Key Idea:
Use $g_{RP}(\cdot) = E\{f(\cdot)|\Phi f(X)\}$ instead of $g(x) = E\{f(x)|f^*\}$, where Φ is some $m \times n$ matrix.
$g_{RP}(\cdot)$ drawn from a GP with covariance kernel

$$q_{RP}(x, z) = (\Phi k_{x,f})^T (\Phi K_{f,f} \Phi^T)^{-1} \Phi k_{f,z},$$

where $k_{x,f} = \{k_{x,x_1}, ..., k_{x,x_n}\}^T$ and $k_{f,z} = \{k_{x_1,z}, ..., k_{x_n,z}\}^T$

Need to apply bias correction to account for variance under-estimation, so approximation drawn from GP with covariance kernel:

$$q_{RM}(x, z) = q_{RP}(x, z) + \delta(x, z)\{k(x, z) - q_{RP}(x, z)\}$$

where $\delta(x, z) = 1$ if $x = z$ and 0 otherwise.

Properties of RP Method

Advantages over "knots" method:

- Lower condition number = better numerical stability
- Lower rank required for a given approximation error

Properties:

- Limiting Case
- Optimality in terms of Hilbert space projection
- Relationship with partial matrix decompositions
- Relationship with truncated series expansions

Woodbury matrix identity

For prediction, we need to calculate $\Sigma_1^{-1} = (Q_{f,f}^{RP} + \sigma^2 I)^{-1}$. Inversion of $Q_{f,f}^{RP}$ is very computationally expensive for large datasets.

Using the Woodbury matrix identity for $Q_{f,f}^{RP} = UD^2U^T$, we have

$$\begin{aligned}
\Sigma_1^{-1} &= (UD^2U^T + \sigma^2 I)^{-1} \\
&= \sigma^{-2} I - \sigma^{-2} U(D^{-2} + \sigma^{-2} U^T U)^{-1} U^T \sigma^{-2} \\
&= \sigma^{-2} I - \sigma^{-4} U(D^{-2} + \sigma^{-2} I)^{-1} U^T
\end{aligned}$$

$D^{-2} + \sigma^{-2} I$ is a diagonal matrix whose inverse is obtained by taking recipricals of the diagonals. **Direct matrix inversion avoided!**

Reduced Rank Matrix Approximation

We want to calculate an optimal projection for our approximation:

- Best approximation is to using a random projection matrix Φ with largest m eigenvalues of K in the SVD:

$$K_m = U_m D_{mm} U_m^T = (\Phi K)^T (\Phi K \Phi^T)^{-1} (\Phi K) \tag{1}$$

- **Problem:** SVD is as expensive as matrix inverse, with $O(n^3)$.
- **Solution:** Approximate spectral decomposition! We can use random $n \times r$ matrix Ω to embed K into $\mathcal{R}^{n \times r}$ space, where $r = \lfloor \frac{m}{\epsilon} \rfloor$, $0 < \epsilon \leq 1$.
  - For low distortion, use Johnson-Lindenstrauss transform, so we populate Ω with $\frac{1}{\sqrt{r}} \omega_{ij}$, where ω_{ij} iid $\sim \mathcal{N}(0, 1)$
  - Now we can find Φ using the Nyström approximation (see Algorithm 1)
- Two considerations:
  - Target rank
  - Target error

Target Rank

Theorem 1

Consider any $0 < \epsilon \leq 1$ and $r = \lfloor \frac{m}{\epsilon} \rfloor$. Obtain K_{tr} from Algorithm 1 for the positive definite matrix K and let K_m be the best rank m approximation for K if terms of $\|\cdot\|_F$. Then,

$$pr\{\|K - K_{tr}\| \leq (1+\epsilon)\|K - K_m\|_F\} \geq \frac{1}{2} \tag{2}$$

Target Rank

Algorithm 1 Approximate spectral decomposition via Nyström method for target rank m

Given a positive definite matrix K of order $n \times n$ and a randomly generated Johnson- Lindenstrauss matrix Ω of order $r \times n$, we find the projection matrix Φ of order $m \times n$ which approximates the range and compute the approximate SVD decomposition via Nyström approximation with Φ.

1. Form the matrix product $K\Omega$.
2. ComputeΦ^T = left factor of the rank $m$spectral projection of the small matrix $K\Omega$.
3. Form $K_1 = \Phi K \Phi^T$.
4. Perform a Choleski factorization of $K_1 = BB^T$.
5. Calculate the Nyström factor $C = K\Phi^T(B^T)^1$.
6. Compute a spectral decomposition for $C = UDV^T$.
7. Calculate the approximate spectral decomposition for $K \approx K_{tr} = UD^2U^T$.

Target Error

Theorem 2

Let $Y = (y_1, y_2, ..., y_n)^T$ be the observed data points and let $\pi_{\text{full}} = \int \pi\{Y, f(X)\} dP\{f(X)\}$, $\pi_{RP} = \int \pi\{Y, g_{RP}(X)\} dP\{g_{RP}(X)\}$ their corresponding marginal distributions. If $\|K_{f,f} - Q^{RP}_{f,f}\|_F \leq \epsilon$, which is the error in approximation of the covariance matrix, then the Kullback Leibler divergence between the marginal distributions from the full and approximated Gaussian process,

$$\text{KL}(\pi_{(full)}, \pi_{RP}) \leq \left\{ n + \left(\frac{n}{\sigma}\right)^2 \right\} \epsilon \tag{3}$$

Target Error

Algorithm 2 Finding range satisfying target error condition

Given a positive definite matrix K of order $n \times n$ and target error $\epsilon > 0$, we find the projection matrix Φ of order $m \times n$ which gives $\|K\Phi^T\Phi K\| < \epsilon$ with probability $1 - \frac{n}{10^r}$.

1. Initialize $j = 0$ and $\Phi = []$, the $0 \times n$ empty matrix.
2. Draw r random vectors $\omega^{(1)}, ..., \omega^{(r)}$ each of order $n \times 1$ with independent entries from $\mathcal{N}(0, 1)$.
3. Compute $\kappa^{(i)} = K\omega^{(i)}$ for $i = 1, ..., r$.
4. Is $max_{i=1,...,r}(\|\kappa^{(j+i)}\|) < \frac{\epsilon\sqrt{\pi}}{10\sqrt{pi}}$? If yes, step 11. If no, step 5.
5. Recompute $j = j + 1, \kappa^{(j)} = [I - \{\Phi^{(j-1)}\}^T\Phi^{(j-1)}]\kappa^{(j)}$ and $\phi^{(j)} = \frac{\kappa^{(j)}}{\|\kappa^{(j)}\|}$.
6. Set $\Phi^{(j)} = \begin{bmatrix} \Phi^{(j-1)} \\ \{\phi^{(j)}\}^T \end{bmatrix}$
7. Draw a $n \times 1$ random vector ω^{j+r} with independent $\mathcal{N}(0, 1)$ entries.
8. Compute $\kappa^{(j+r)} = [I\{\Phi^{(j)}\}^T\Phi^{(j)}]K\omega^{(j+r)}$.
9. Recompute $\kappa^{(i)} = \kappa^{(i)}\phi^{(j)}\langle\phi^{(j)}, \kappa^{(i)}\rangle$ for $i = (j + 1), ..., (j + r1)$.
10. Back to target error check in step 4.
11. Output $\Phi = \Phi^{(j)}$.

Experimental Results

For $m = 10$	$\|\|\|_F$	$\|\|\|_2$	Cond No	Time
RP	106.1377	17.6578	1.0556	0.06
PP1	107.6423	17.6776	1.2356	0.04
PP2	106.6644	17.6778	1.2619	0.04

For $m = 25$	$\|\|\|_F$	$\|\|\|_2$	Cond No	Time
RP	82.1550	17.2420	1.7902	0.22
PP1	91.1016	17.5460	230.236	0.18
PP2	85.6616	17.3800	13.8971	0.21

For $m = 50$	$\|\|\|_F$	$\|\|\|_2$	Cond No	Time
RP	50.5356	14.2998	2.9338	0.27
PP1	79.1030	17.0172	2803.5	0.24
PP2	69.5681	15.6815	876.23	0.25

For $m = 100$	$\|\|\|_F$	$\|\|\|_2$	Cond No	Time
RP (Algo1)	6.6119	2.8383	20.6504	0.40
PP1	39.9642	13.1961	1.3815×10^6	0.31
PP2	10.1639	6.3082	1792.1	0.36

Table 1: Comparative performance of the approximations in terms of matrix error norms, with the random projection approach based on Algorithm 1.

Experimental Results

		PP1	PP2	RP
$n = 100, \epsilon = 0.1$, optimal m = 5	Required Rank	17	9	7
	Cond No	298.10	54.59	20.08
	Time	0.03	0.04	0.07
$n = 1000, \epsilon = 0.01$, optimal m = 69	Required Rank	213	97	78
	Cond No	2.30×10^7	2164.6	473.43
	Time	12.1	11.5	36.2
$n = 10000, \epsilon = 0.01$, optimal m = 137	Required Rank	1757	793	174
	Cond No	3.19×10^{19}	2.30×10^9	1012.3
	Time	335	286	214

Table 2: Comparison of the ranks required to achieve specific target errors by the different algorithms, with random projection based on Algorithm 2

Experimental Results

		PP1	PP2	RP
$\epsilon = 0.1$, smooth	MSPE	11.985	8.447	3.643
	Avg Required Rank	1715.6	453.8	117.2
	95% Interval Required Rank	[1331,2542]	[377,525]	[97,141]
	Posterior Mean, θ_1	0.09	0.10	0.06
	95% credible interval, θ_1	[0.05,0.14]	[0.05,0.15]	[0.04,0.08]
	ESS, θ_1	496	870	1949
	Posterior Mean, θ_2	0.91	1.15	1.25
	95% credible interval, θ_2	[0.58,1.58]	[0.85,1.43]	[1.09,1.46]
	ESS, θ_2	2941	3922	4518
	Avg ESS, Predicted Values	2190	3131	5377
	Time	39761	29355	32365
$\epsilon = 0.01$, wavy	MSPE	10.114	6.891	2.265
	Avg Required Rank	3927.1	941.5	129.7
	95% Interval Required Rank	[2351,5739]	[868,1165]	[103,159]
	Posterior Mean, θ_1	0.07	0.08	0.13
	95% credible interval, θ_1	[0.01,0.14]	[0.02,0.15]	[0.09,0.17]
	ESS, θ_1	574	631	1918
	Posterior Mean, θ_2	0.83	0.85	0.79
	95% credible interval, θ_2	[0.21,1.74]	[0.40.1.63]	[0.45,1.29]
	ESS, θ_2	3679	4819	5002
	Avg ESS, Predicted Values	2875	3781	5769
	Time	78812	47642	33799
$\epsilon = 0.01$, very wavy	MSPE	17.41	13.82	6.93
	Avg Required Rank	4758.5	1412.5	404.5
	95% Interval Required Rank	[2871,6781]	[1247,1672]	[312,475]
	Posterior Mean, θ_1	0.11	0.09	0.05
	95% credible interval, θ_1	[0.04,0.17]	[0.05,0.13]	[0.03,0.08]
	ESS, θ_1	741	747	1049
	Posterior Mean, θ_2	1.27	1.18	1.19
	95% credible interval, θ_2	[1.08,1.43]	[1.12,1.41]	[1.15,1.34]
	ESS, θ_2	1521	2410	2651
	Avg ESS, Predicted Values	1263	1415	2422
	Time	89715	57812	47261

Table 3: Simulated data sets with the target error algorithm for the three different simulations. Different algorithms compared in terms of preditive MSE and various posterior summaries for the unknown parameters. ESS stands for effective sample size.

Code Example

Let's try it!