Automatic online tuning for fast Gaussian summation
Vlad I. Morariu, Balaji V. Srinivasan, Vikas C. Raykar, Ramani Duraiswami, Larry S. Davis

Problem
\[
G(y_j) = \sum_{i=1}^{N} q_i \exp \left( -\frac{||y_j - y_i||^2}{h^2} \right), \quad j = 1, \ldots, M
\]
- Direct computation is quadratic O(MN).
- Slows down kernel machines.
- Algorithms such as FGT [1] and IFGT [2] obtain speedup by approximating to specified error ε, but do not perform well across all bandwidths h, point distributions, and error ε and need tuning.

Our solution
- Uses the IFGT and a tree data structure.
- Four methods, each optimal in different situations.
- Black box approach: Automatically predicts fastest method and tunes its parameters for the given dataset.
- Compares favorably to Dual-Tree methods [3].

Improved Fast Gauss Transform
- Reduces cost of Gauss Transform to O(N+M) [2].
- Uses truncated Taylor expansion of exponential term
  \[
e^{-\frac{\|x-y\|^2}{h^2}} \approx \sum_{k=0}^{p-1} \frac{(-1)^k}{k!} \left( \frac{\|x-y\|^2}{h^2} \right)^k + \Delta_{ij}
\]
- The error of truncating series starting at pth term is bounded by
  \[
  \Delta_{ij} \leq \frac{p}{p!} \frac{\|x_j-x_i\|^p}{h^p} \left( \frac{\|x_j-x_i\|^p}{h^p} \right)^p e^{-\frac{\|x_j-x_i\|^2}{h^2}}
\]
- Truncation number is chosen for each source point by assuming worst case placement of a target point and choosing p s.t.
  \[
  \Delta_{ij} \leq \varepsilon \quad \text{which guarantees that} \quad |g(y_j) - g(y_i)|/Q \leq \varepsilon
\]
  where Q is the sum of all q_i.

Summary of approach:
1) Choose number of centers K and max truncation \( p_{\text{max}} \)
2) Cluster source points using K-center algorithm
3) Pre-compute contributions from sources
   \[
   C_{\text{src}} = \sum_{y_i \in S_k} q_i e^{-\frac{\|y_i-x_i\|^2}{h^2}} \left( \frac{y_i-x_i}{h} \right)^\alpha \quad \text{for} \ \alpha \leq p_{\text{max}} - 1
   \]
4) For each target \( y_j \), evaluate by collecting contributions from clusters within cut-off radius
   \[
   g(y_j) = \sum_{\|x_j-x_i\| \leq \varepsilon} \sum_{\|y_i-x_i\| \leq \varepsilon} C_{\text{src}} e^{-\frac{\|y_j-y_i\|^2}{h^2}} \left( \frac{y_j-y_i}{h} \right)^\alpha
   \]

Problems:
- Parameters are optimized for uniform distributions.
- IFGT performs poorly for low bandwidths where kernels are local.

Automatic IFGT parameter tuning
Choosing number of clusters without assuming a uniform distribution
- Instead of estimating cluster radius as \( k^{1/2} \), use incremental clustering to evaluate actual cluster radius. Thus choosing optimal K based on actual distribution.

Choosing cluster-wise truncation numbers
- Allow some points to have higher error as long as other points in the same cluster have low enough error to compensate.
- Ensure that for each cluster \( S_k \)
  \[
  \sum_{y_j \in S_k} |g_j| \Delta_{ij} \leq \sum_{y_j \in S_k} |g_j| \varepsilon = Q_k \varepsilon
  \]

Tree data structure
- Using a tree data structure, can compute Gauss Transform efficiently across bandwidths.
- Must choose one of four evaluation methods.

Direct: evaluate equation directly; works well for small N and M, and does not have high overhead cost.

Direct+Tree: build tree directly on sources; works well for small bandwidths.

IFGT: works well for large N and M, and medium to large bandwidths.

IFGT+Tree: build tree on cluster centers; works well for case where number of clusters K is large.

Comparison of methods

Automatic method selection
- \( n_s \) and \( n_c \) are the average number of sources and cluster centers, respectively, that are within the cut-off radius of a query target.
- \( n_s \) and \( n_c \) can be approximated from sub-sampled dataset.
- Given \( d, N, M, K, n_s \), and \( n_c \), we can estimate cost of each method.

Summary of method selection approach:
1) Estimate \( n_s \).
2) Calculate \( \text{Cost}_{\text{IFGT}}(d, N, M) \) and \( \text{Cost}_{\text{IFGT+Tree}}(d, N, M, n_c) \).
3) Estimate highest \( K_{\text{max}} \) for which \( \text{fgt} \) and \( \text{fgt+tree} \) could be faster.
4) If \( K_{\text{max}} > 0 \)
   a) Compute IFGT parameters \( K \) and \( n_{c_{\text{max}}} \).
b) Estimate \( n_s \); estimate \( \text{Cost}_{\text{IFGT}} \) and \( \text{Cost}_{\text{IFGT+Tree}} \).
5) Return \( \text{arg min} \{ \text{Cost} \} \).

Results
Comparison with Dual-Tree approach

Gaussian Process Regression

References

Open source C/C++ and MATLAB bindings
http://sourceforge.net/projects/figtree