The problem: big datasets

Could be large:

\( N \) (#data), \( D \) (#features), \( M \) (#models)
What we’d like

Allow users to apply all the state-of-the-art statistical methods…

….with orders-of-magnitude more computational efficiency

Core methods of statistics / machine learning / mining

- **Querying**: nearest-neighbor $O(N)$, spherical range-search $O(N)$, orthogonal range-search $O(N)$, contingency table
- **Density estimation**: kernel density estimation $O(N^2)$, mixture of Gaussians $O(N)$
- **Regression**: linear regression $O(D^3)$, kernel regression $O(N^2)$, Gaussian process regression $O(N^3)$
- **Classification**: nearest-neighbor classifier $O(N^2)$, nonparametric Bayes classifier $O(N^2)$, support vector machine
- **Dimension reduction**: principal component analysis $O(D^3)$, non-negative matrix factorization, kernel PCA $O(N^3)$, maximum variance unfolding $O(N^3)$
- **Outlier detection**: by robust $L_2$ estimation, by density estimation, by dimension reduction
- **Clustering**: $k$-means $O(N)$, hierarchical clustering $O(N^3)$, by dimension reduction
- **Time series analysis**: Kalman filter $O(D^3)$, hidden Markov model, trajectory tracking
- **2-sample testing**: $n$-point correlation $O(N^n)$
- **Cross-match**: bipartite matching $O(N^3)$
5 main computational bottlenecks:
Aggregations, GNPs, graphical models, linear algebra, optimization

- Querying: nearest-neighbor $O(N)$, spherical range-search $O(N)$, orthogonal range-search $O(N)$, contingency table
- Density estimation: kernel density estimation $O(N^2)$, mixture of Gaussians $O(N)$
- Regression: linear regression $O(D^3)$, kernel regression $O(N^2)$, Gaussian process regression $O(N^3)$
- Classification: nearest-neighbor classifier $O(N^2)$, nonparametric Bayes classifier $O(N^2)$, support vector machine
- Dimension reduction: principal component analysis $O(D^3)$, non-negative matrix factorization, kernel PCA $O(N^3)$, maximum variance unfolding $O(N^3)$
- Outlier detection: by robust L$_2$ estimation, by density estimation, by dimension reduction
- Clustering: k-means $O(N)$, hierarchical clustering $O(N^3)$, by dimension reduction
- Time series analysis: Kalman filter $O(D^3)$, hidden Markov model, trajectory tracking
- 2-sample testing: n-point correlation $O(N^n)$
- Cross-match: bipartite matching $O(N^3)$

How can we compute these efficiently?

**Multi-scale Decompositions**

e.g. kd-trees

[Bentley 1975], [Friedman, Bentley & Finkel 1977],[Moore & Lee 1995]
How can we compute these efficiently?

- Generalized N-body algorithms (multiple trees) for distance/similarity-based computations [2000, 2003, 2009]
- Multi-scale Monte Carlo for linear algebra and summations [2007, 2008]
- Stochastic process approximations for time series [2009]
- Monte Carlo optimization: online, progressive [2009]

Computational complexity using fast algorithms

- Querying: nearest-neighbor $O(\log N)$, spherical range-search $O(\log N)$, orthogonal range-search $O(\log N)$, contingency table
- Density estimation: kernel density estimation $O(N)$ or $O(1)$, mixture of Gaussians $O(\log N)$
- Regression: linear regression $O(D)$ or $O(1)$, kernel regression $O(N)$ or $O(1)$, Gaussian process regression $O(N)$ or $O(1)$
- Classification: nearest-neighbor classifier $O(N)$, nonparametric Bayes classifier $O(N)$, support vector machine $O(N)$
- Dimension reduction: principal component analysis $O(D)$ or $O(1)$, non-negative matrix factorization, kernel PCA $O(N)$ or $O(1)$, maximum variance unfolding $O(N)$
- Outlier detection: by robust $L_2$ estimation, by density estimation, by dimension reduction
- Clustering: k-means $O(\log N)$, hierarchical clustering $O(N\log N)$, by dimension reduction
- Time series analysis: Kalman filter $O(D)$ or $O(1)$, hidden Markov model, trajectory tracking
- 2-sample testing: n-point correlation $O(N^{\log p})$
- Cross-match: bipartite matching $O(N)$ or $O(1)$
Computational complexity using fast algorithms

- **Querying:** nearest-neighbor $O(\log N)$, spherical range-search $O(\log N)$, orthogonal range-search $O(\log N)$, contingency table
- **Density estimation:** kernel density estimation $O(N)$ or $O(1)$, mixture of Gaussians $O(\log N)$
- **Regression:** linear regression $O(D)$ or $O(1)$, kernel regression $O(N)$ or $O(1)$, Gaussian process regression $O(N)$ or $O(1)$
- **Classification:** nearest-neighbor classifier $O(N)$, nonparametric Bayes classifier $O(N)$, support vector machine $O(N)$
- **Dimension reduction:** principal component analysis $O(D)$ or $O(1)$, non-negative matrix factorization, kernel PCA $O(N)$ or $O(1)$, maximum variance unfolding $O(N)$
- **Outlier detection:** by robust $L_2$ estimation, by density estimation, by dimension reduction
- **Clustering:** $k$-means $O(\log N)$, hierarchical clustering $O(N \log N)$, by dimension reduction
- **Time series analysis:** Kalman filter $O(D)$ or $O(1)$, hidden Markov model, trajectory tracking
- **2-sample testing:** $n$-point correlation $O(N \log n)$
- **Cross-match:** bipartite matching $O(N)$ or $O(1)$
Ex: 3-point correlation runtime

(biggest previous: 20K)

VIRGO simulation data,
N = 75,000,000

naïve: 5x10^9 sec.
(~150 years)

multi-tree: 55 sec.
(exact)

Ex: support vector machine

Data: IJCNN1 [DP01a]
2 classes
49,990 training points
91,701 testing points
22 features

SMO: 12,831 SV’s, 84,360 iterations, 98.3% accuracy, 765 sec

SFW: 4,145 SV’s, 4,500 iterations, 98.1% accuracy, 21 sec
Software

• MLPACK (C++)
  – First scalable comprehensive ML library

• MLPACK-db
  – Fast data analytics in relational databases (SQL Server)

• MLPACK Pro
  - Very-large-scale data

Issues

• How to disseminate/integrate?
• In-database/centralized or not?
• Trust of complex algorithms?
• Other statistical/ML needs?