VisWeek 09
VIS • INFOVIS • VAST

SAND2009-6641C

ParaView

Statistics

Philippe Pébay
David Thompson
Janine Bennett
Diana Roe
Nathan Fabian

Sandia National Laboratories

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94-AL85000

Outline

- Statistics in General
- Statistics in VTK
- Statistics in ParaView
- Algorithm Details

VTK Filters

Tasks

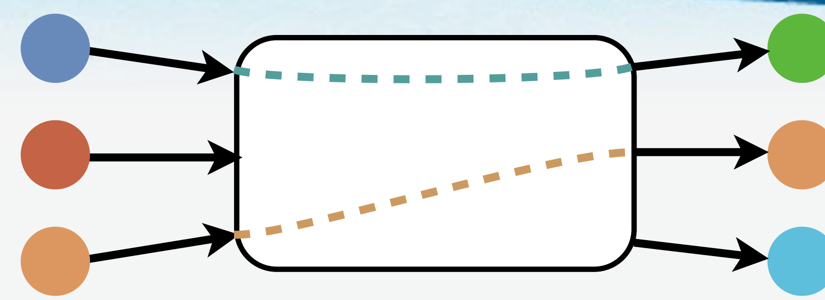
- **Learn** from input data. Also called **Train** in the machine learning/classification community.
- **Derive** further (related and/or more user-accessible) information from minimal statistics.
- **Appraise** the model; detect
  - problems with assumptions (independence, goodness of fit); and
  - stability problems (numerical & sensitivity).
- **Assess** some data using what was learned.

Design Pattern

With distributed data, most statistics algorithms look like trendy applications of

- **Learn** – Map-Reduce
- **Derive** – Embarrassingly Parallel Reduce
- **Appraise** – Map-Reduce
- **Assess** – Embarrassingly Parallel Map

VTK Statistics



- Filters have **inputs** for
  - Data to learn or assess
  - Model parameters (e.g., k-means start points)
  - Pre-existing model for assessment
- Filters have **outputs** for
  - Possibly-assessed data
  - Model output
  - Assessment summary information

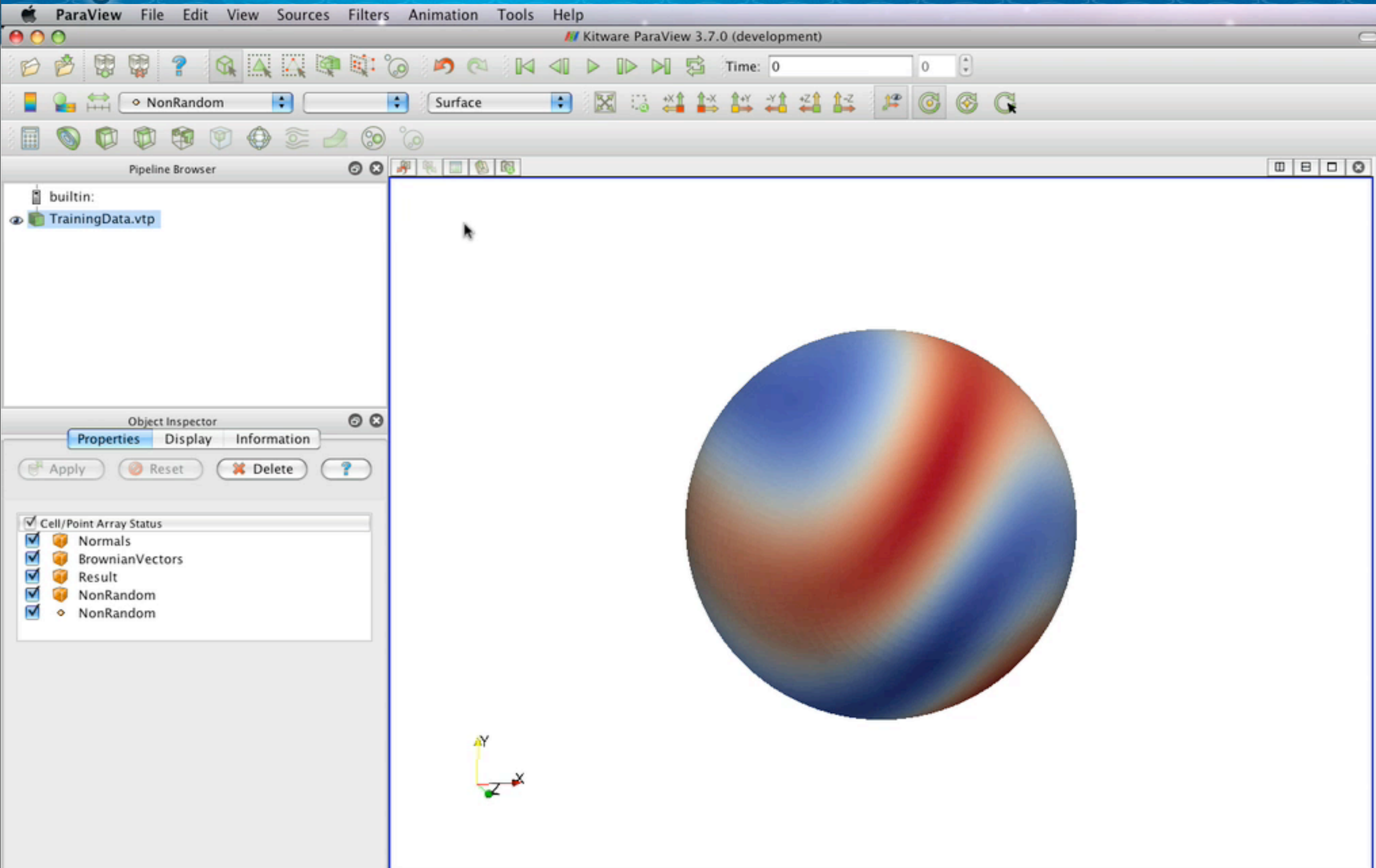
VTK Statistics

- Filters include
  - Contingency tables
  - Descriptive statistics
  - *k*-means clustering
  - Order statistics (quantiles)
  - Principal component analysis
  - Bivariate histogram (for parallel coords)
- Currently no filters implement Appraise but all implement Learn, Derive, & Assess.

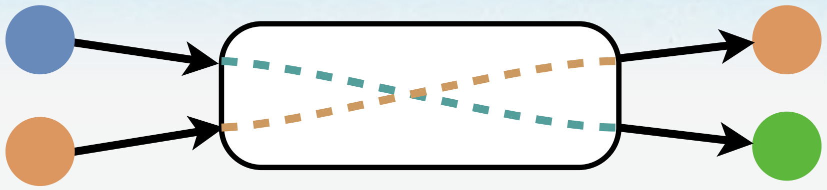
★ Filters in blue have parallel implementations.

ParaView Interface

ParaView Statistics



ParaView Statistics



- Filters have **inputs** for
  - Data to learn or assess
  - Pre-existing model for assessment
- Filters have **outputs** for
  - Model output
  - Possibly-assessed data
- Notice reversed output order (for ease of use)!

Statistics Caveats

- ⚠ In data-parallel mode, **point** arrays will have **distorted** statistics: shared points are counted once per process instead of just once.
- ⚠ Distortion may be introduced by your mesh (spatially varying sampling frequency).
- ⚠ Tasks that perform random sampling will choose a **different** random sample each time the filter is re-executed.

Algorithm Details

Details: Contingency

Learn + Derive

- Counts number of occurrences of all combinations of values
- Marginalizes with respect to each array component
- Computes information entropies

Details: Contingency

Assess

- Assigns probability from contingency table to each observation.
- Computes Pointwise Mutual Information (PMI) of each observation.
- Note that when you Learn from a different dataset or a subset of the data, any values not encountered during Learn will be assessed with 0 probability. This can make the output look noisy.

Details: Descriptive

Learn

- Computes the min, max, mean, and M2–M4 centered sums.

Derive

- Adds columns for standard deviation, variance, and estimators for skewness and kurtosis.

Assess

- Tags each observation with signed (or unsigned) number of deviations from the mean.

Details: k-means

Learn

- Iteratively updates k cluster centers x_i until maximum count or relative tolerance met.
- Initial x_i are taken from a uniform random distribution over each array's bounds **or** a third input table for model parameters.

Derive

- Compares total error of each (k,x_i) set to determine lowest-error fit. (Not useful in ParaView: only a single value of k is allowed.)

Details: k-means

Derive, *cont.*

- Use in VTK allows comparisons between multiple k values and initial cluster centers.

Assess

- Tags each observation with 2 values:
  - Integer ID of nearest cluster center
  - Distance to cluster center (Euclidean)

Details: Multicorrelative

Learn

- Computes means of arrays and covariances of array pairs

Derive

- Computes Cholesky decomposition of the covariance matrix (used in **Assess**).

Assess

- Uses the inverse of the covariance matrix to tag each observation with its Mahalanobis distance.

Details: Multicorrelative

- Output table is densely packed with multiple matrices and vectors.
- Covariance matrix is symmetric; only the top half is stored.
- Cholesky decomposition is lower-triangular.
- Overall: N+1 × N+1 table for N arrays.

	Column	Mean	BrownianVectors_0	BrownianVectors_1	BrownianVectors_2	Result
0	BrownianVectors_0	0.0130061	0.0903729	-0.00155543	0.00117395	0.000430427
1	BrownianVectors_1	0.0202801	0.300621	0.0863474	0.00163257	-0.00264618
2	BrownianVectors_2	-0.00266763	-0.00517405	0.293804	0.0905124	-0.0040427
3	Result	0.00479249	0.00390508	0.00562544	0.300775	0.0898239
4	Cholesky	1587	0.00143179	-0.00898141	-0.0132915	0.299273

Details: Multicorrelative

- Output table is densely packed with multiple matrices and vectors.
- Covariance matrix is symmetric; only the top half is stored.
- Cholesky decomposition is lower-triangular.
- Overall: N+1 × N+1 table for N arrays.

	Column	Mean	BrownianVectors_0	BrownianVectors_1	BrownianVectors_2	Result
0	BrownianVectors_0	0.0130061	0.0903729	-0.00155543	0.00117395	0.000430427
1	BrownianVectors_1	0.0202801	0.300621	0.0863474	0.00163257	-0.00264618
2	BrownianVectors_2	-0.00266763	-0.00517405	0.293804	0.0905124	-0.0040427
3	Result	0.00479249	0.00390508	0.00562544	0.300775	0.0898239
4	Cholesky	1	0.00143179	-0.00898141	-0.0132915	0.299273

Mean

Covariance

Cholesky decomposition

#Vals

Details: PCA

Learn

- Identical to multicorrelative statistics

Derive

- Optionally normalizes covariance matrix, then computes SVD to get eigenanalysis.

Assess

- Projects each observation into the new basis, which may be truncated to a fixed dimension or a fixed "energy."

Details: PCA

- Output table is densely packed with multiple matrices and vectors.
- Multicorrelative output is identical but without the final N+1 rows.

	Column		Mean	BrownianVectors_0	BrownianVectors_1	BrownianVectors_2	Result
0	BrownianVectors_0		0.0130061	0.0903729	-0.00155543	0.00117395	0.000430427
1	BrownianVectors_1		0.0202801	0.300621	0.0863474	0.00163257	-0.00264618
2	BrownianVectors_2		-0.00266763	-0.00517405	0.293804	0.0905124	-0.0040427
3	Result		0.00479249	0.00390508	0.00562544	0.300775	0.0898239
4	Cholesky		1587	0.00143179	-0.00898141	-0.0132915	0.299273
5	PCA	0	1.06379	-0.0652366	0.490468	0.582203	-0.645156
6	PCA	1	1.01444	0.826499	-0.411326	0.380697	-0.052727
7	PCA	2	0.970223	-0.518189	-0.76089	0.262885	-0.288821
8	PCA	3	0.951554	-0.210058	0.106293	0.668581	0.705391
9	PCA	Cov	0	0.0903729	0.0863474	0.0905124	0.0898239

Details: PCA

- Output table is densely packed with multiple matrices and vectors.
- Multicorrelative output is identical but without the final N+1 rows.

	Column		Mean	BrownianVectors_0	BrownianVectors_1	BrownianVectors_2	Result
0	BrownianVectors_0		0.0130061	0.0903729	-0.00155543	0.00117395	0.000430427
1	BrownianVectors_1		0.0202801	0.300621	0.0863474	0.00163257	-0.00264618
2	BrownianVectors_2		-0.00266763	-0.00517405	0.293804	0.0905124	-0.0040427
3	Result		0.00479249	0.00390508	0.00562544	0.300775	0.0898239
4	Cholesky		1[illegible]	0.00143179	-0.00898141	-0.0132915	0.299273
5	PCA	0	1.06379	-0.0652366	0.490468	0.582203	-0.645156
6	PCA	1	1.01444	0.826499	-0.411326	0.380697	-0.052727
7	PCA	2	0.970223	-0.518189	-0.76089	0.262885	-0.288821
8	PCA	3	0.951554	-0.210058	0.106293	0.668581	0.705391
9	PCA	Cov	[illegible]	0.09037[illegible]	[illegible]	[illegible]	[illegible]98239

Covariance

Mean

Cholesky decomposition

#Vals

Eigen-values

Eigenvectors (row vectors)

Unused

Eigenvector normalization