A Universal Postprocessing Toolkit for Accelerator Simulation and Analysis

Michael Borland

Operations Analysis Group

Advanced Photon Source

Argonne National Laboratory

Presented at the
1998 International Computational Accelerator Physics Conference
Monterey, California

Common Problems with Current Simulation Codes

- The user's postprocessing effort is generally linear in the number of runs performed.
- Sharing data between codes is labor-intensive and fragile.
- Users frequently resort to modifying a published code to get what they need.

Why Are These Problems Prevalent?

For most codes:

- single-code postprocessor or none at all
- most output is not compute-ready as far as the user is concerned
- data file protocols are code-specific and fragile

A Solution is Available

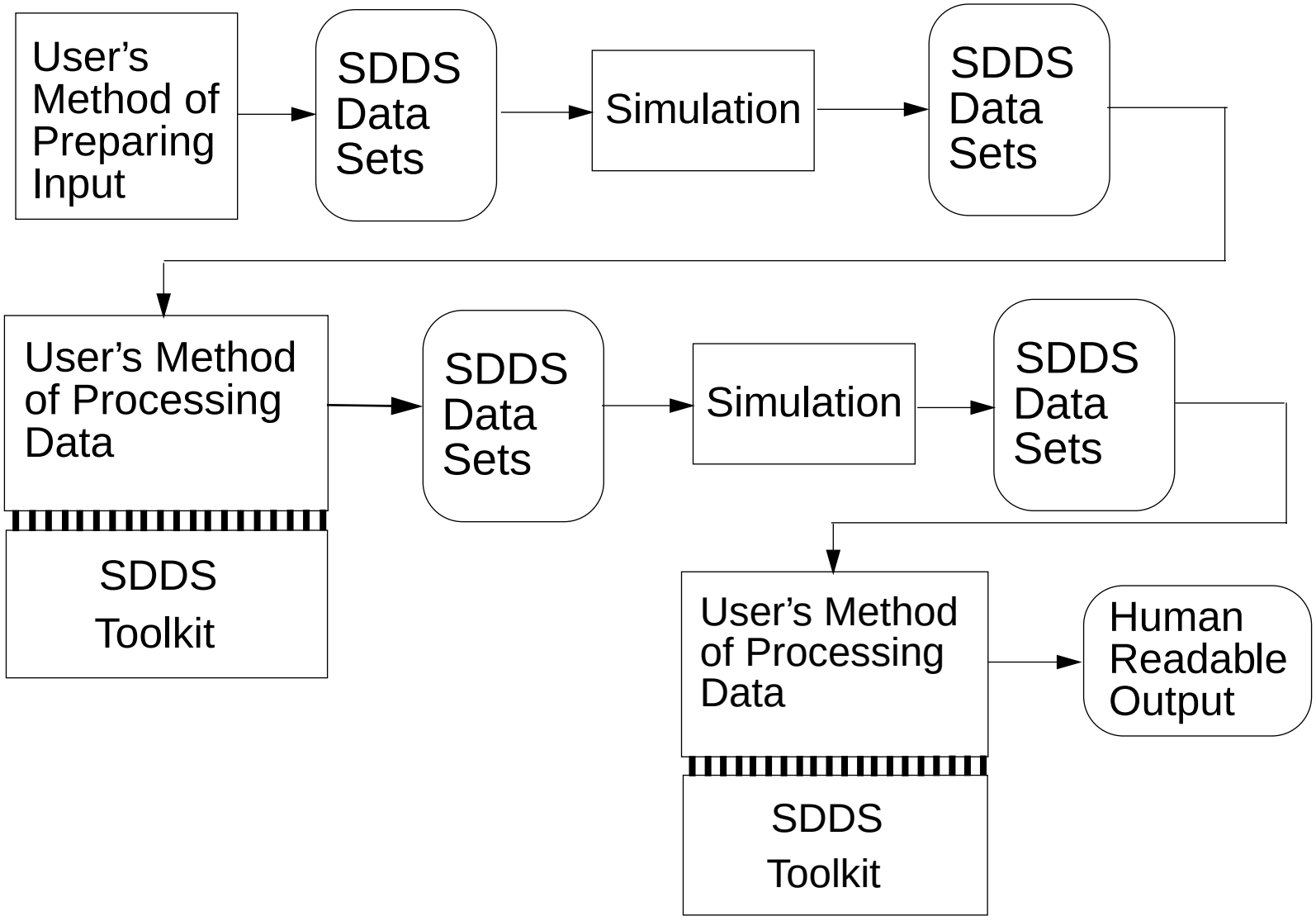
The *Self-Describing Data Sets* (SDDS) system provides a common pre- and post-processing system for simulations.

It includes

- a relatively simple but very useful self-describing file protocol
- a toolkit of ~70 general-purpose programs
- additional toolkit of ~20 control-system programs (for EPICS)

Operations Analysis Group
Michael Borland borland@aps.anl.gov

Conceptual Example of SDDS-Linked Simulations



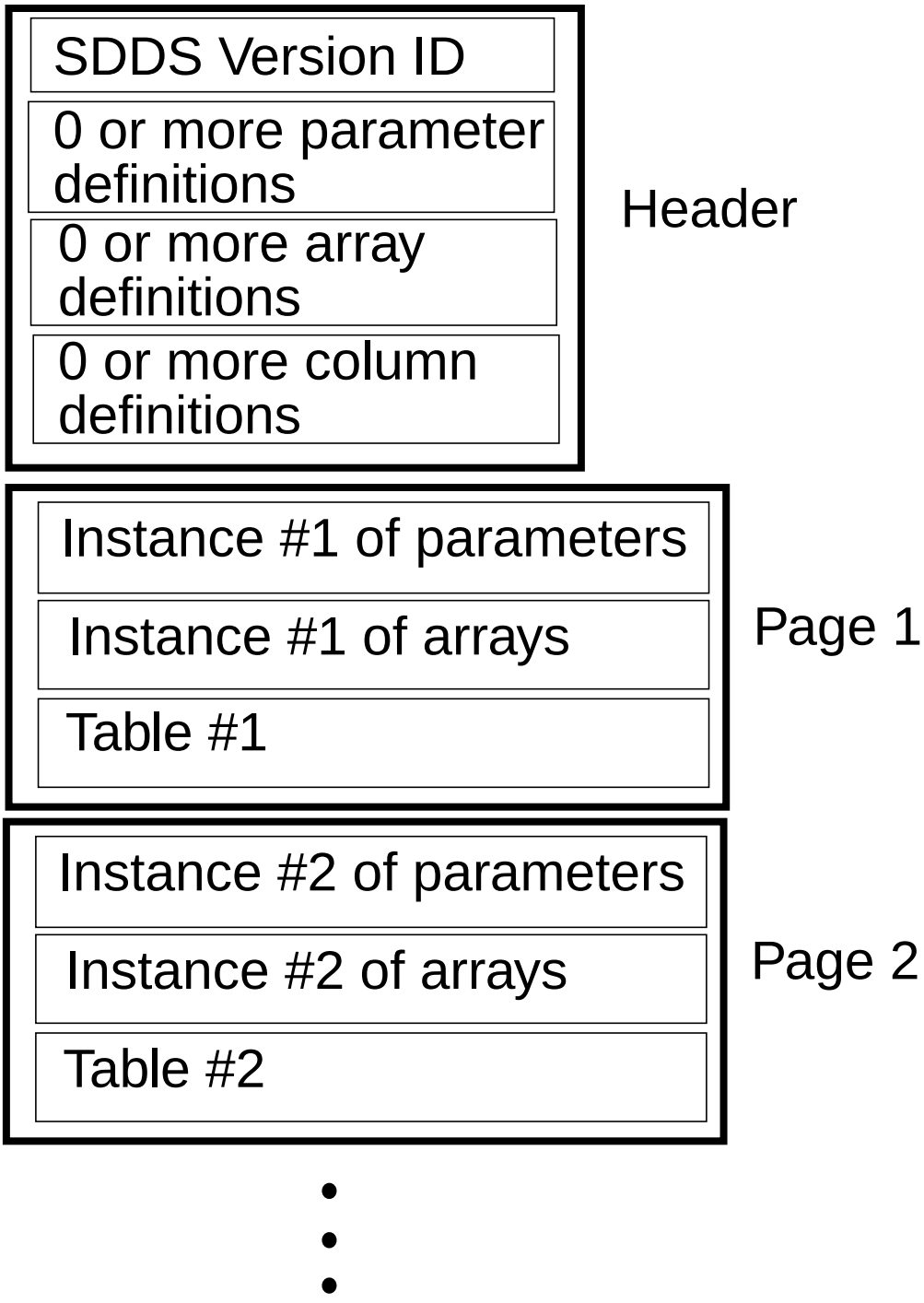
Wide Application of SDDS

- Used for commissioning of APS ring and its injector starting in February 1994. Chosen over MATLAB, IDL, GUS, etc.
- Used for
  - accelerator simulation (14 codes)
  - accelerator high level applications, data logging, data review, etc.
  - experimental data collection and analysis
- SDDS has never caused operational downtime.

SDDS File Protocol

- Data is identified and accessed by name only, using a library of C routines.
- Files include meta-data about data, e.g., units and data type.
- Data may be ASCII or binary.
- No limits on the size or number of data elements.
- Simple data model.

SDDS Data Model



Some Advantages of Using SDDS Files

- Programs can’t be broken by the addition of new data.
- Allows robust input programming.
- Data type flexibility.
- Data is self-documenting.
- Programs can be generic and operate on *named* data.

Program Toolkits

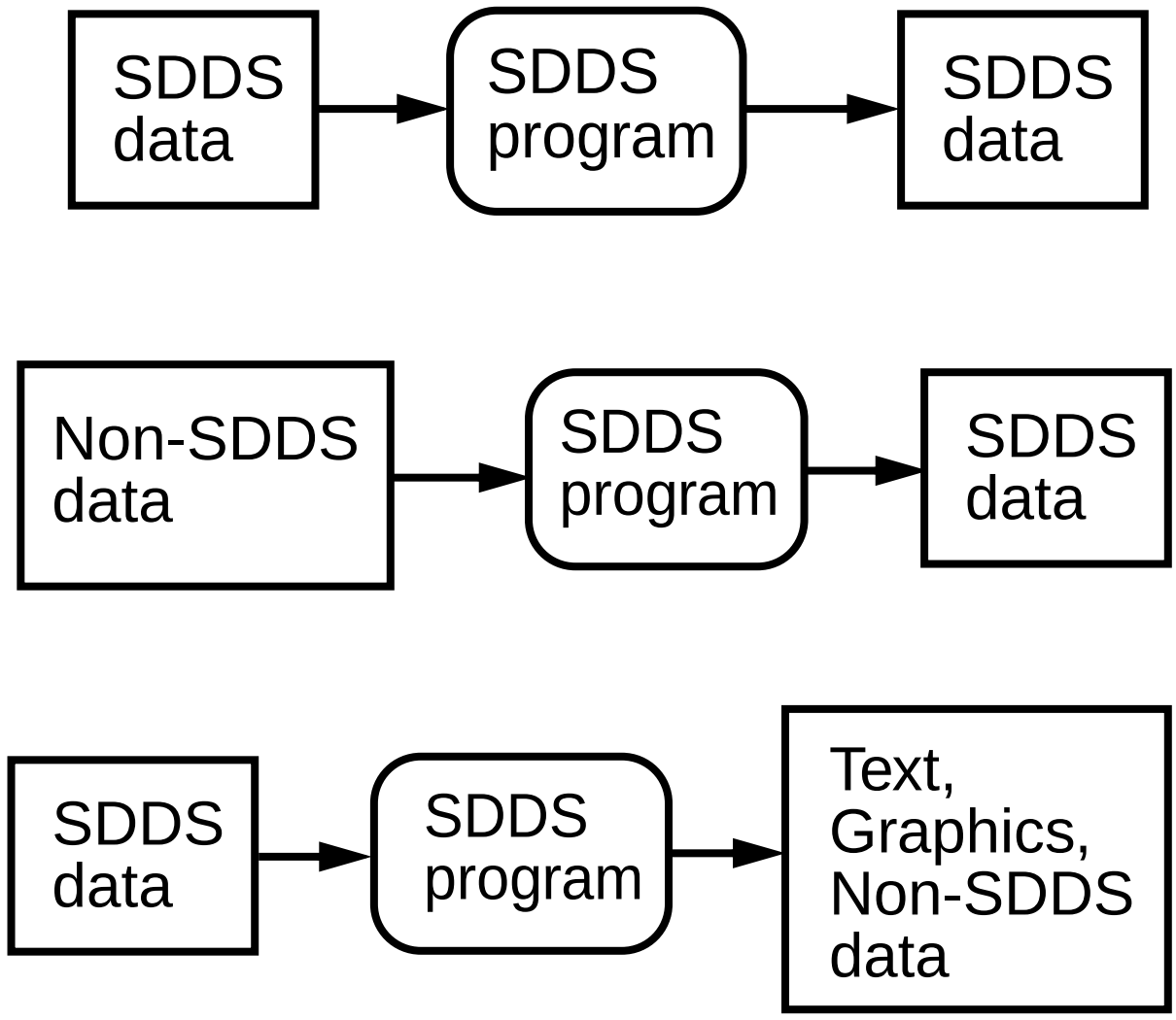
A “toolkit” is a group of programs that can operate on the same type of entity.

- Such programs are generic, configurable, and relatively simple.
- One program’s output is another’s input.
- Decentralized expansion:
  - anyone can add to it independently
  - additions *can’t* break existing tools or applications
- Enhancement doesn’t involve debugging a single, large program.

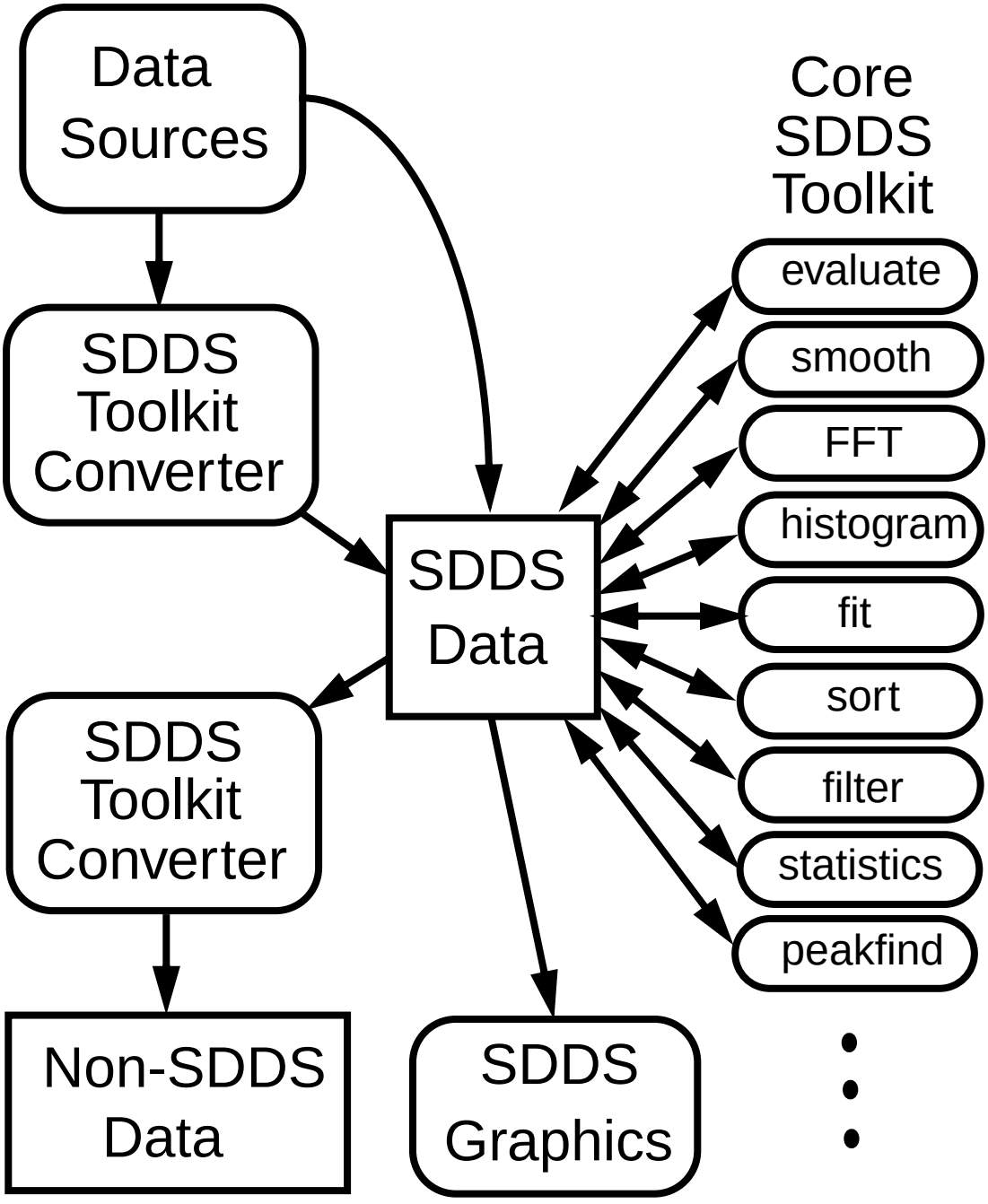
Some SDDS Toolkit Capabilities

- Device-independent graphics.
- Equation evaluation.
- Data winnowing.
- Statistics and histograms.
- Polynomial, exponential, and gaussian fitting.
- Correlation and outlier analysis.
- Matrix operations (e.g., SVD).
- Cross-referencing, sorting, and collation.
- FFTs, digital filtering, and convolution.
- Smoothing and interpolation.
- Peakfinding.
- Protocol conversion to/from SDDS.
- Text printouts of data.

Three Types of SDDS Toolkit Programs



SDDS Toolkit Paradigm



Operations Analysis Group
Michael Borland borland@aps.anl.gov

SDDS and Programming

- Using SDDS is not like programming.
- For example, to take an FFT of V vs t:

  sddsfft *input output* -column=t,V
  sddsplot -column=f,FFTV *output*

- Take the FFT of BPM00, BPM01,..., BPM99, then compute the mean PSD:

  sddsfft *input* -pipe=out -column=t,BPM?? -psdOutput \
  | sddsrowstats -pipe=in *output* -mean=MeanPSD,PSDBPM??
  sddsplot -column=f,MeanPSD *output*

Operations Analysis Group
Michael Borland borland@aps.anl.gov

- Same, but integrate square-root of PSD vs frequency:

```
sddsfft input -pipe=out -column=t,BPM?? -psdOutput \
| sddsrowstats -pipe  mean=MeanPSD,PSDBPM?? \
| sddsprocess -pipe -define=column,RMSMotion,"MeanPSD sqrt" \
| sddsinteg -pipe=in output -integrate=RMSMotion -versus=f
sddsplot -column=f,RMSMotionInteg output
```

- Find maximum value and its position for each FFT:

```
sddsfft input -pipe=out -column=t,BPM?? \
| sddssmooth -pipe -columns=FFTBPM* -despike \
| sddsprocess -pipe -process=BPM??,max,%sMax \
  -process=BPM??,max,%sPos,functionOf=f,position \
| sddscollapse -pipe \
| sddscollect -pipe=in output -collect=suffix=Max -collect=suffix=Pos
sddsplot -column=Rootname,Max output
```

Scripts

- SDDS tools are command-line driven, so they are scriptable.
  SDDS adds computational muscle to scripts.
- At APS, Tcl/Tk is used for complex work.
- Advantages of using SDDS and scripts
  - an easy way to create and debug complex processing
  - provides a record of what was done
  - effortless repetition of analysis for new data
  - easy modification of previous analysis methods

SDDS-Compliant Simulation Codes

These codes emit only SDDS files, use only SDDS for postprocessing, and also accept SDDS files as input.

- **elegant** — General-purpose accelerator code.[1]
- **shower** — EGS4 wrapper for electron-gamma shower simulation.[2]
- **efield** — Used with shower for multilayer target simulations.[3]
- **clinchor** — Computes coupled bunch growth rates.[2]
- **spiffe** — Fully-electromagnetic PIC code for rf gun simulation.[1]
- **rfgun** — rf gun design code.[1]

1. M. Borland.
2. L. Emery.
3. E. Lessner

Use of SDDS Files by elegant

- Input/output:
  - particle coordinates
- Input:
  - input of data for any element parameter
  - input of impedances and wake functions
  - input of cavity and energy ramps
  - input of kicker waveforms
- Output:
  - turn-by-turn particle data (coordinates or statistics)
  - FFTs of particle motion
  - beam moments vs s
  - transport matrix vs s
  - Twiss parameters vs s
  - lattice parameters (tunes, chromaticities, etc.)
  - coordinates of lost particles
  - initial coordinates of transmitted particles
  - final beam parameters (size, energy, emittance, etc.)
  - amplification factors
  - orbits, corrector strengths, and statistics
  - magnet strengths after tune and chromaticity correction
  - output of internally-generated error data

Use of SDDS Files by spiffe

- Input/output
  - electromagnetic fields
- Input
  - antennae excitation current
  - cathode temporal emission profile
- Output
  - particle coordinates
  - particle snapshots
  - field samples in time and space
  - solenoid fields

Complex Data in a Simple File Protocol

Using “parallel files,” we can store complex data in a relatively simple file protocol. This is why **elegant** and **spiffe** can use SDDS for all of their diverse output.

- Programs create one file for each type of output.
- Usually, related files have similar names.
- Generally, a specific page in one file goes with the same page in all related files.
- Cross-referencing and merging tools are available.

Output-Complaint Codes

These codes write but don't read SDDS files:

- **capture_spc** — Simulation of longitudinal motion with space charge.[1]
- **ABCI*** — Cavity-beam interaction code.[2]
- **TDA3D*** — FEL code.[3]

*APS version.

1. E. Lessner.
2. Y-H Chin. Modified by Y-C Chae.
3. T-M Tran and J. S. Wurtele. Modified by Y-C Chae.

Codes with External Output Conversion to SDDS

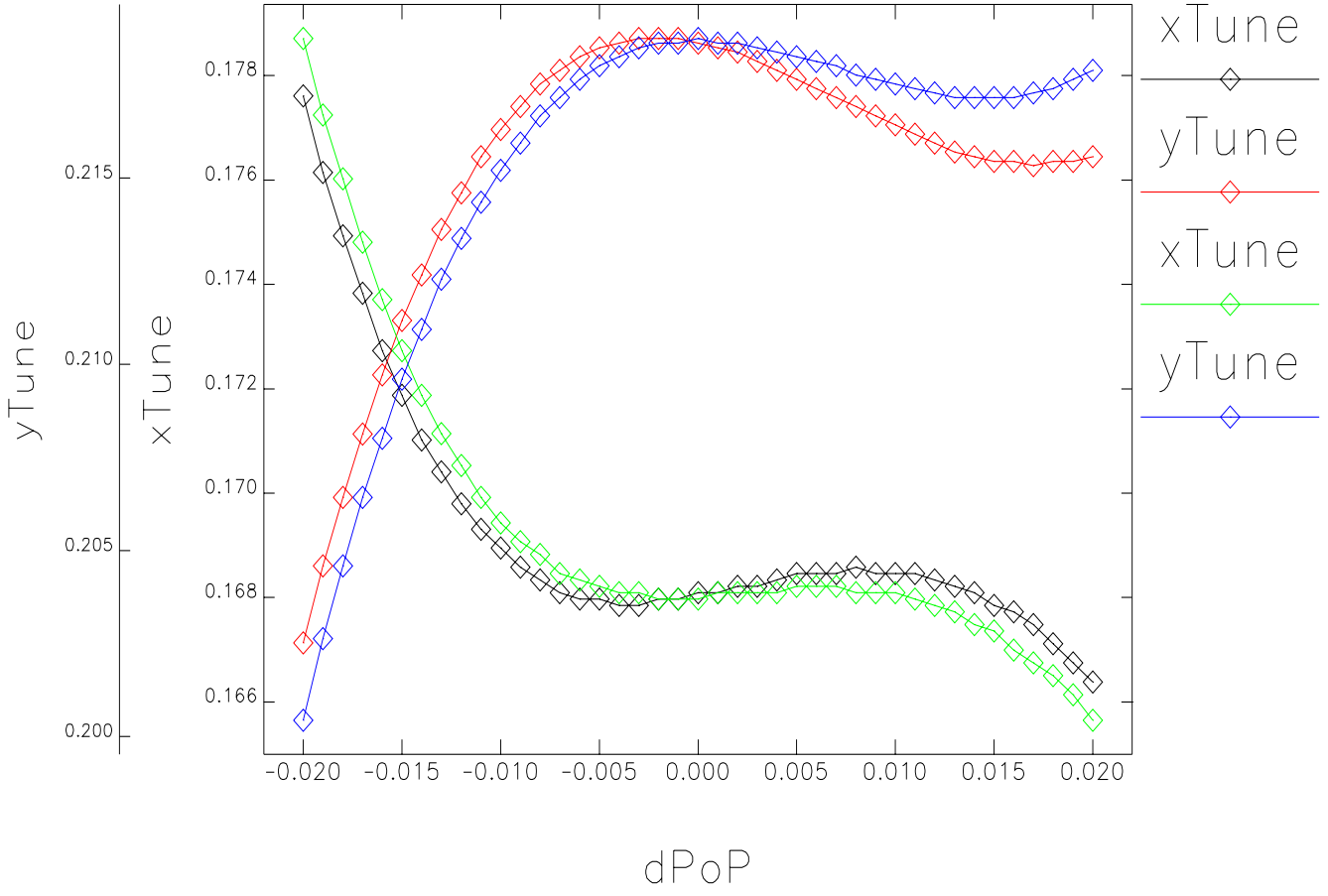
- **MAD** — General-purpose accelerator code.[1]
- **MAFIA** — 3D cavity simulation code.[2]
- **ACCSIM** — Tracking code.[3]
- **RACETRACK** — Tracking code.[4]
- **GINGER** — FEL code.[5]

1. H. Grote and F. C. Iselin. TFS to SDDS converter by M. Borland.
2. T. Weiland. MAFIA to SDDS converter by L. Emery.
3. TRIUMF code. Conversion header by Y-C Chae.
4. A. Wrulich. Conversion header by Y-C Chae.
5. W. M. Fawley. Conversion header by Y-C Chae.

Example: Enhacing a Code Using SDDS

- The program **elegant** does canonical tracking, but doesn't provide tune vs $\frac{\Delta P}{P}$.
- With SDDS, we can get this information without modifying the code:
  - prepare input particle coordinates having range of $(\Delta P)/P$ values
  - run **elegant**
  - FFT turn-by-turn x and y coordinates
  - find tunes by finding peaks of FFTs
  - associate tunes with $(\Delta P)/P$ values
- Can also do chromaticity matching from tracking results.

Operations Analysis Group
Michael Borland borland@aps.anl.gov



```
#!/bin/csh
# A sample csh script using SDDS tools to find tune variation
# with energy from turn-by-turn tracking data.

# Make input particles with dP/P from -0.02 to 0.02,
# one particle per SDDS page.
sddssequence -pipe=out -define=dPoP0,type=double \
  -sequence=begin=-0.02,end=0.02,delta=0.001 \
  | sddsbreak -pipe -rowlimit=1 \
  | sddsprocess -pipe=in "-define=column,p,880 dPoP0 1 + *" \
  -define=column,t,0,units=s -define=column,xp,0 -define=column,yp,0 \
  -define=column,x,0,units=m -define=column,y,0,units=m \
  -define=parameter,dPoP,dPoP0 \
  par.input

elegant par.ele    # track the particles

# FFT the turn-by-turn data
# associate with the dP/P value from file made above
# find the locations of the peaks (=tunes)
sddsfft par.turnByTurn -pipe=out -column=Pass,x,y \
  -suppressAverage -window=Hanning \
  | sddsxref -pipe par.input -transfer=parameter,dPoP -leave=* \
  | sddsprocess -pipe \
  -process=FFTx,max,xTune,functionOf=f,position \
  -process=FFTy,max,yTune,functionOf=f,position \
  | sddscollapse -pipe=in par.tune
# Plot the results
sddsplot -column=dPoP,?Tune par.tune -yScales=names \
   -graph=symbol,vary=subtype,scale=2,connect=subtype -legend
```

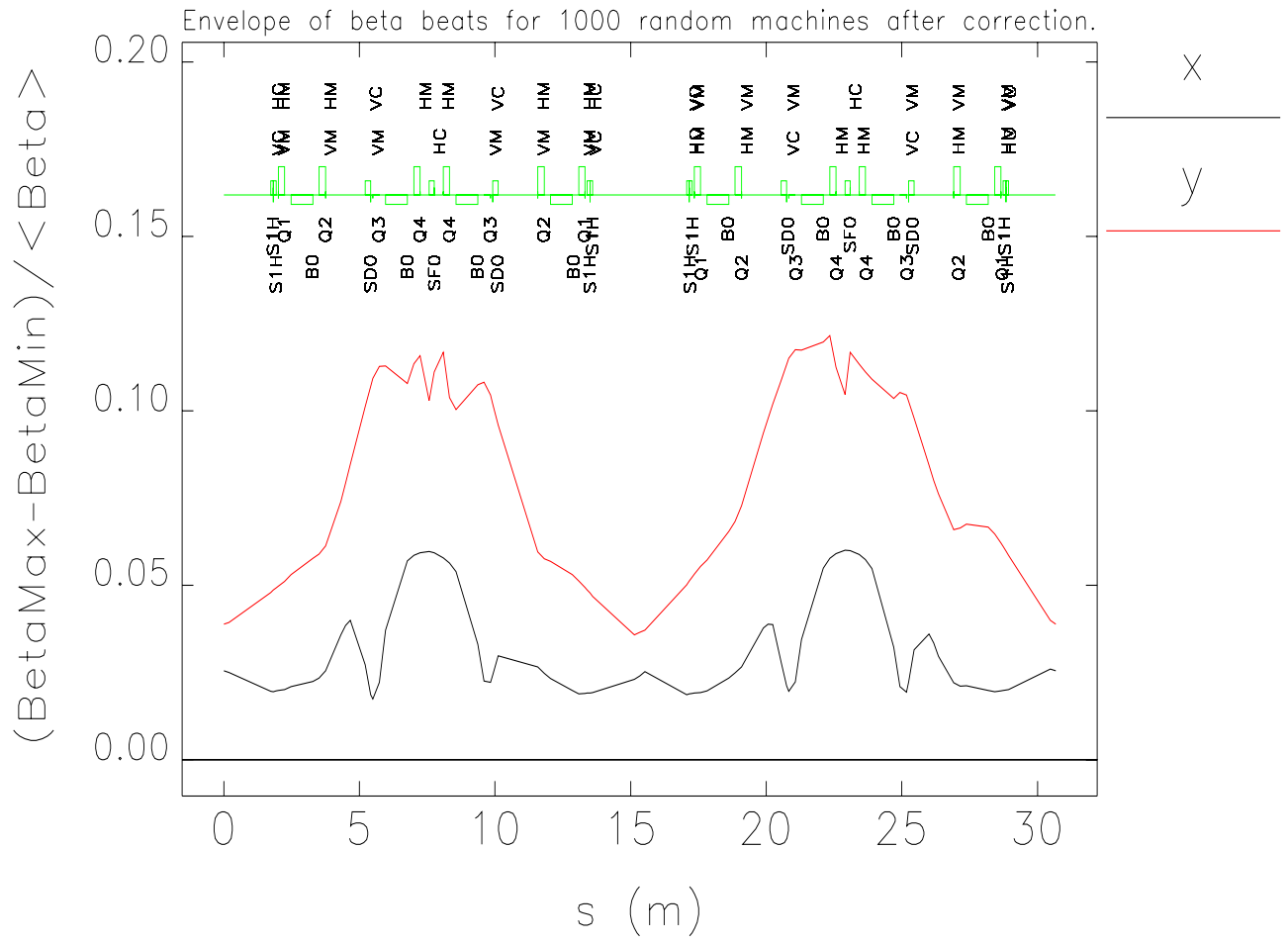
Operations Analysis Group
Michael Borland borland@aps.anl.gov

Example: Postprocessing 1000 Runs

- Want PAR lattice functions with
  - magnet alignment and strength errors
  - orbit, tune, and chromaticity correction
- Using 20 separate workstations, 1000 random sets takes ~15 minutes.
- An SDDS-based script does all data analysis and display.
- Easy to repeat for a new lattice.
- Similar applications:
  - dynamic aperture
  - injection efficiency

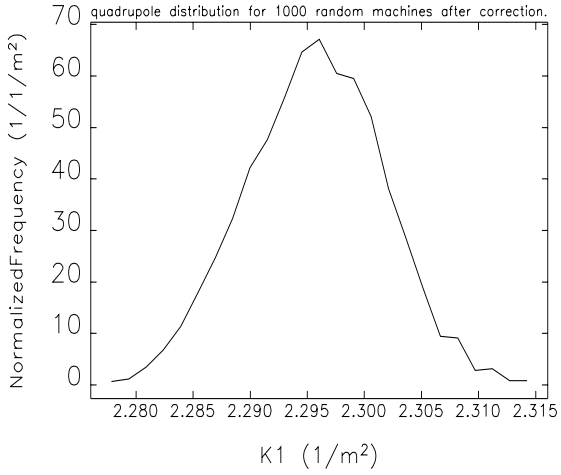
Operations Analysis Group
Michael Borland borland@aps.anl.gov

Beta beat for 1000 Random Machines

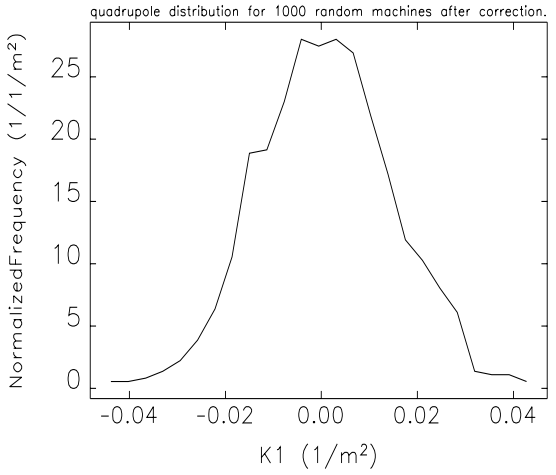


Operations Analysis Group
Michael Borland borland@aps.anl.gov

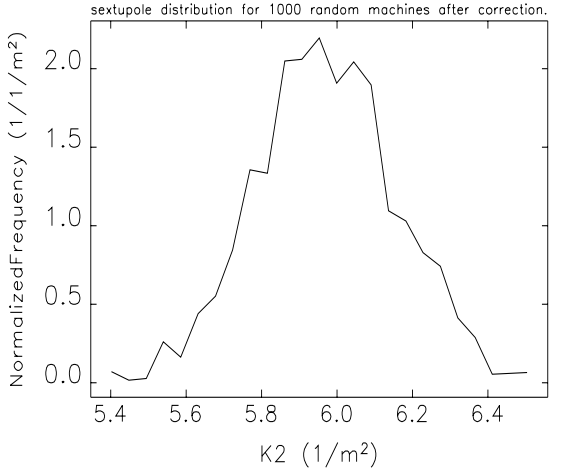
Statistics of Quads and Sexts for 1000 Machines



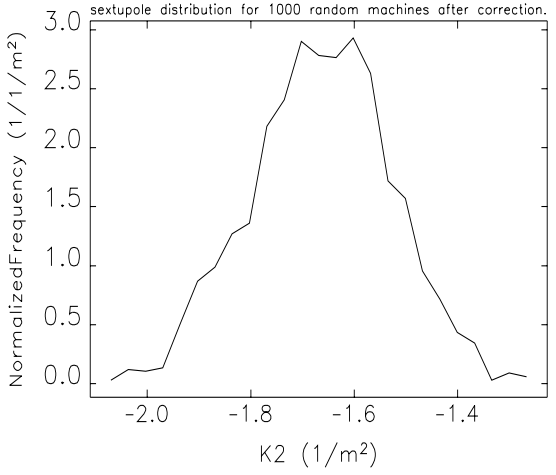
Q2 statistics Mean: 2.296 StDev: 0.006



Q3 statistics Mean: 0.001 StDev: 0.014



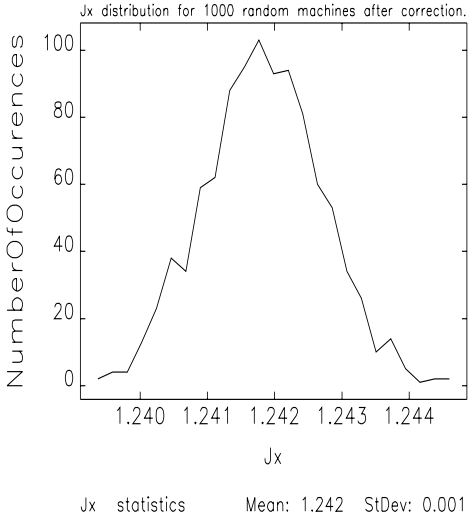
Mean: 5.970 StDev: 0.186



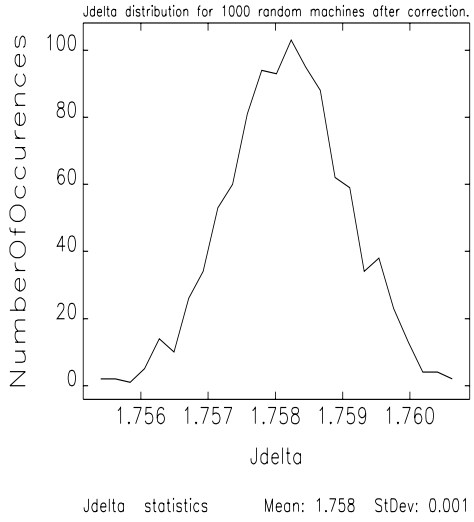
Mean: -1.664 StDev: 0.135

Operations Analysis Group
Michael Borland borland@aps.anl.gov

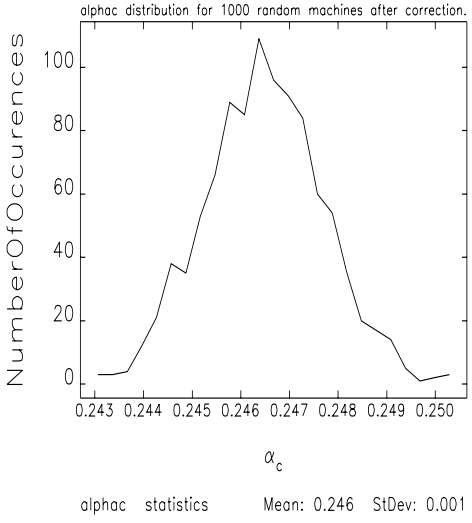
Statistics of Some Global Parameters for 1000 Machines



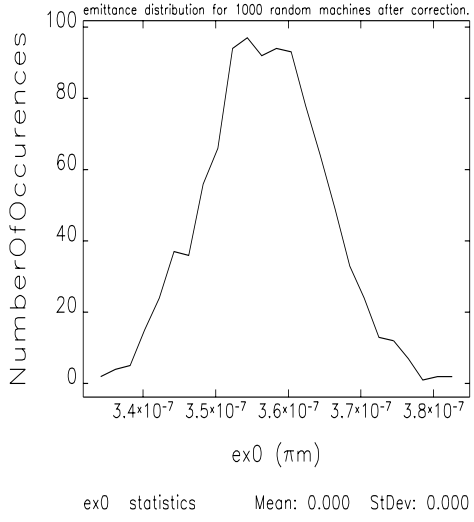
Jx statistics Mean: 1.242 StDev: 0.001



Jdelta statistics Mean: 1.758 StDev: 0.001



alphac statistics Mean: 0.246 StDev: 0.001



ex0 statistics Mean: 0.000 StDev: 0.000

Multi-Code Simulations Using SDDS

- **shower** and **efield:** multilayer positron target for slow positron production.[1]
- **spiffe** and **elegant:** design and optimization of APS rf gun, transport line, and linac.[2]
- **shower** and **elegant*:*** positron emission and capture in APS linac.
- **shower** and **elegant:** effects of window on injected beam losses**.**

1. M. White and E. Lessner, “Slow Positron Target Concepts for the APS Linear Accelerator,” Proc. 11th International Conf. on Positron Annihilation, Material Sciences Forum Volumes 255-257, pp. 778-780 (1997).
2. M. Borland, “An Improved Thermionic Microwave Gun and Emittance-Preserving Transport Line”, Proc 1993 PAC, pp 3015-3017. Also J. Lewellen, private communication.

Top-Up Safety Tracking

- Top-up operation, wherein beam is injected with shutters open, is a high-priority goal at APS.
- Issue:
  Assuming a partial dipole short, can beam be extracted down a photon beamline during injection while keeping stored beam?
- Approach:
  Evaluate survival of stored beam and escape of injected beam for various fault scenarios.
- Work of M. Borland and L. Emery for APS top-up safety review.

—Overview of Top-Up Simulations—

- Each aperture configuration requires 542 runs of **elegant**.
- 12 groups of runs simulate different failure scenarios.
- Use ~20 workstations, taking ~2 days.
- Tcl scripts set up the runs. Most runs are
  - set up from templates
  - configured using SDDS files, often from another **elegant** run in the set

Top-up Tracking Setup Data

- Match 22 different integer tunes for APS. Quad settings are saved as SDDS files.
- For selected lattices, find stable range of variation for a single quadrupole of each family and save the data in SDDS files.
- Generate acceptance-filling phase-space distributions for two types of photon beamline and save as SDDS files.
- Use SDDS toolkit to create 2D field maps for quadrupoles and sextupoles from PE2D simulation output.

Top-Up Simulation —Stored Beam Survival—

- Compute closed orbit for various lattices and various fault conditions. Evaluate beam survival with:
  - dipole strength error,
  - plus single quadrupole errors within the stable range of tunes,
  - plus single quadrupole having a maximally dangerous dipole field due to a short.
- SDDS files used for magnet strengths and quadrupole error range.
- For each scenario, find dipole strength error at which beam at which beam no longer survives.

Top-Up Simulation —Photon Beamline Tracking—

- Track backwards down photon beamlines with conditions corresponding to the matching closed orbit simulation.
- SDDS used for
  - reading data computed during the stored beam simulations. For example, may load quadrupole strength and dipole kick values.
  - input particle distributions from previous tracking in photon beamlines.
  - field maps of quadrupoles and sextupoles giving fields at large x.
- For each scenario, find the dipole strength error at which no beam exits the sector.

Operations Analysis Group
Michael Borland borland@aps.anl.gov

Top-Up Simulation —Postprocessing—

- There is no manual data gathering or calculation required.
- A postprocessing script using the SDDS toolkit is used for each group of simulations. These may be used individually to review results.
- Another script runs the group scripts to produce final results.
- From 130MB of data, we get a *single number* and a *one-word message* that tells us if top-up is safe under the simulated conditions, e.g.:

```
Minimum FSE gap: 0.0502   Safe!
```

- An SDDS file is also produced with more detailed summary data.

Operations Analysis Group
Michael Borland borland@aps.anl.gov

Conclusion

- The SDDS data file protocol is suitable for use with diverse codes.
- The SDDS toolkit provides powerful, general pre- and post-processing for diverse codes.
- Advantages of this approach:
  - scriptable pre- and post-processing for compliant codes
  - combined use of codes is easy
  - allows highly complex, automated use and extension of codes