Sparc/Solaris to PC/Linux

- Capabilities of PC/Linux chip sets
- Software Porting
- Byte Ordering
- Precision
Capabilities of PC/Linux chip sets

Are PC/Linux chips capable of precise scientific calculations?

- **Floating Point Benchmark** – Tests execution speed & accuracy using a floating point intensive real world application on an algorithm that is very sensitive to errors.

- **Fast Fourier Transform Benchmark** – a fast Fourier transform of a square matrix of complex numbers followed by the inverse transform

- **Correct matching results on PC/Linux and Sparc/Solaris using any of gcc, cc, f77, g77, ifort, etc.**

- [http://www.fourmilab.ch/fbench](http://www.fourmilab.ch/fbench)
Software Porting

• To test porting, we took the GONG++ Ring Pipeline, about 17,000 lines of C and Fortran, and created the Portable Ring Pipeline, a single set of source code that automatically configures and compiles on both Sparc/Solaris and PC/Linux.

• Uses different compilers on each platform.

• Produced with the GNU autoconf(1) “configurator”

• Sparc/Solaris vs. PC/Linux results are well within the error bars, but still show differences within floating point precision. Although the differences are negligible, we can learn what kinds of precision differences to expect between platforms.
Software Porting

Sparc/Solaris vs. PC/Linux comparisons of zonal velocities (top) and meridional components (bottom) measured on a patch near the limb using the Portable Ring Pipeline. Each graph plots two sets of results: Sparc/Solaris vs. PC/Linux.
Byte Ordering

• Big Endian vs. Little Endian

• Portable Ring Pipeline installs proper inversion kernel

• Use of CFITSIO package helps

• Rice, our custom compression tool, modified to always handle Big Endian (Solaris) files even when running on Linux
  so that all historical (Solaris) files will uncompress on Linux.
  Automatic platform independent configuration and compilation
done using GNU autoconf(1).
Precision

• We know it is possible to get the exact same results: ffbench

• Much of our current software (IRAF, GRASP, etc) uses floating point (single precision) for mathematical calculations

• Continual use of single precision allows errors to creep within the 6th decimal place, making comparisons between Solaris and Linux difficult

• Some experience suggests that PC/Linux will generate more accurate results using the current code than Solaris.

• Long term, we want all floating point math to be done with double precision all the way through, and then converted to single precision just before writing the results out.