

Reconstructing fast ion distribution functions from NUBEAM

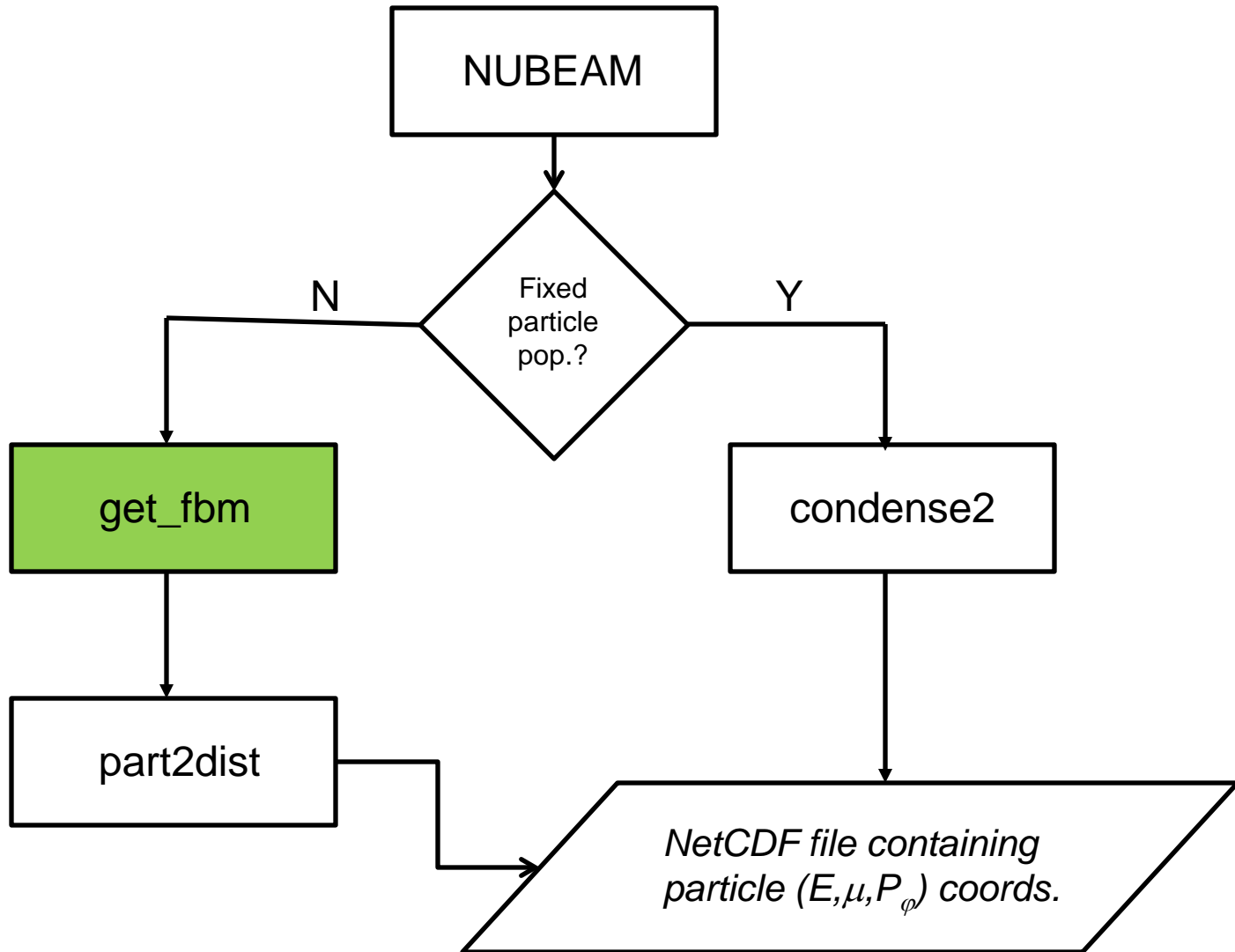
Joshua Breslau and Deyong Liu

TRANSP Users Group Meeting
May 4, 2017

Motivation

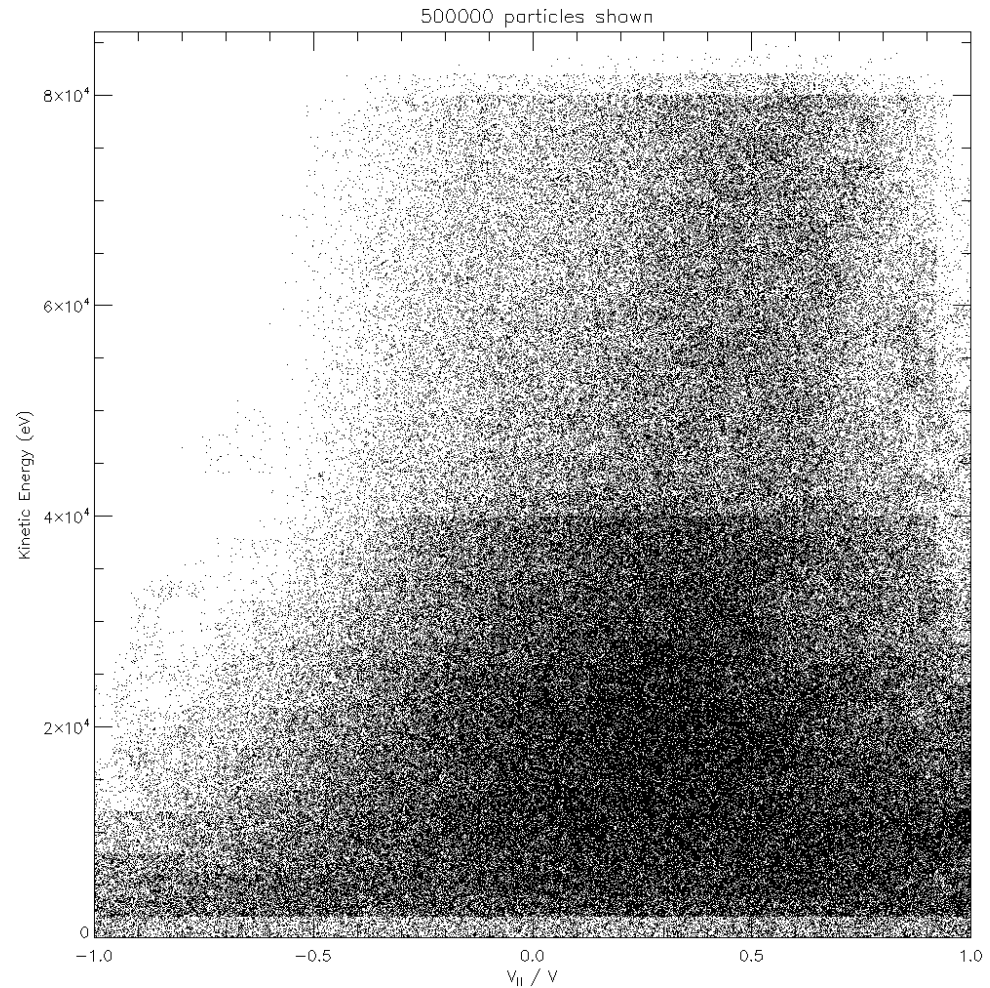
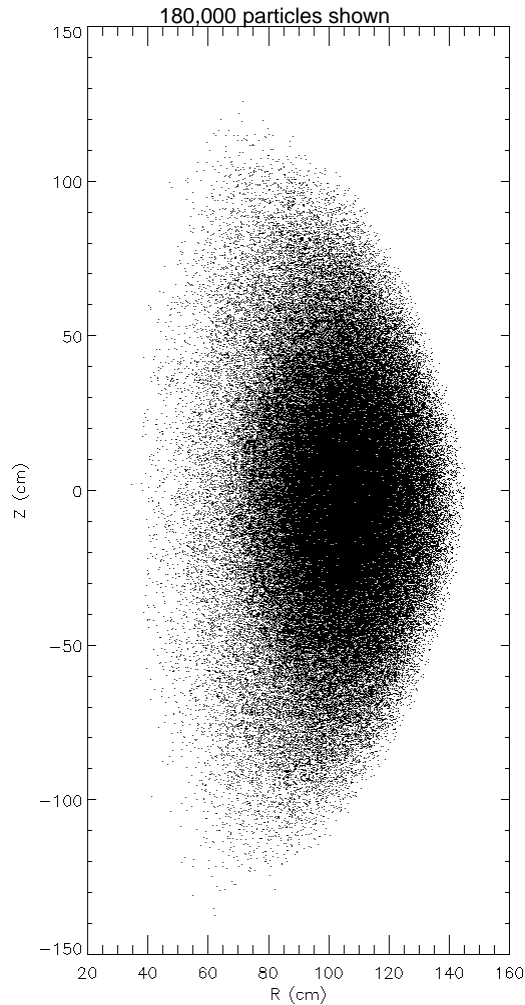
- Various kinetic stability codes would benefit from the use of realistic hot ion distribution functions from TRANSP NUBEAM Monte Carlo calculations, e.g.,
 - NOVA-K computes the modification to ideal linear MHD growth rate due to the energetic particle population.
 - M3D-K combines a gyrokinetic δf energetic particle advance with an extended MHD fluid advance for the bulk plasma in nonlinear time-dependent hybrid simulations.
- These codes are typically initialized with analytic particle distributions with a small number of free parameters.
- The distribution function must be specified in terms of three constants of the motion, with the fourth dimension eliminated. For δf calculations, the function must be smooth enough to allow derivatives to be taken with respect to energy and toroidal angular momentum.

Workflow, pt. 1: get particle data

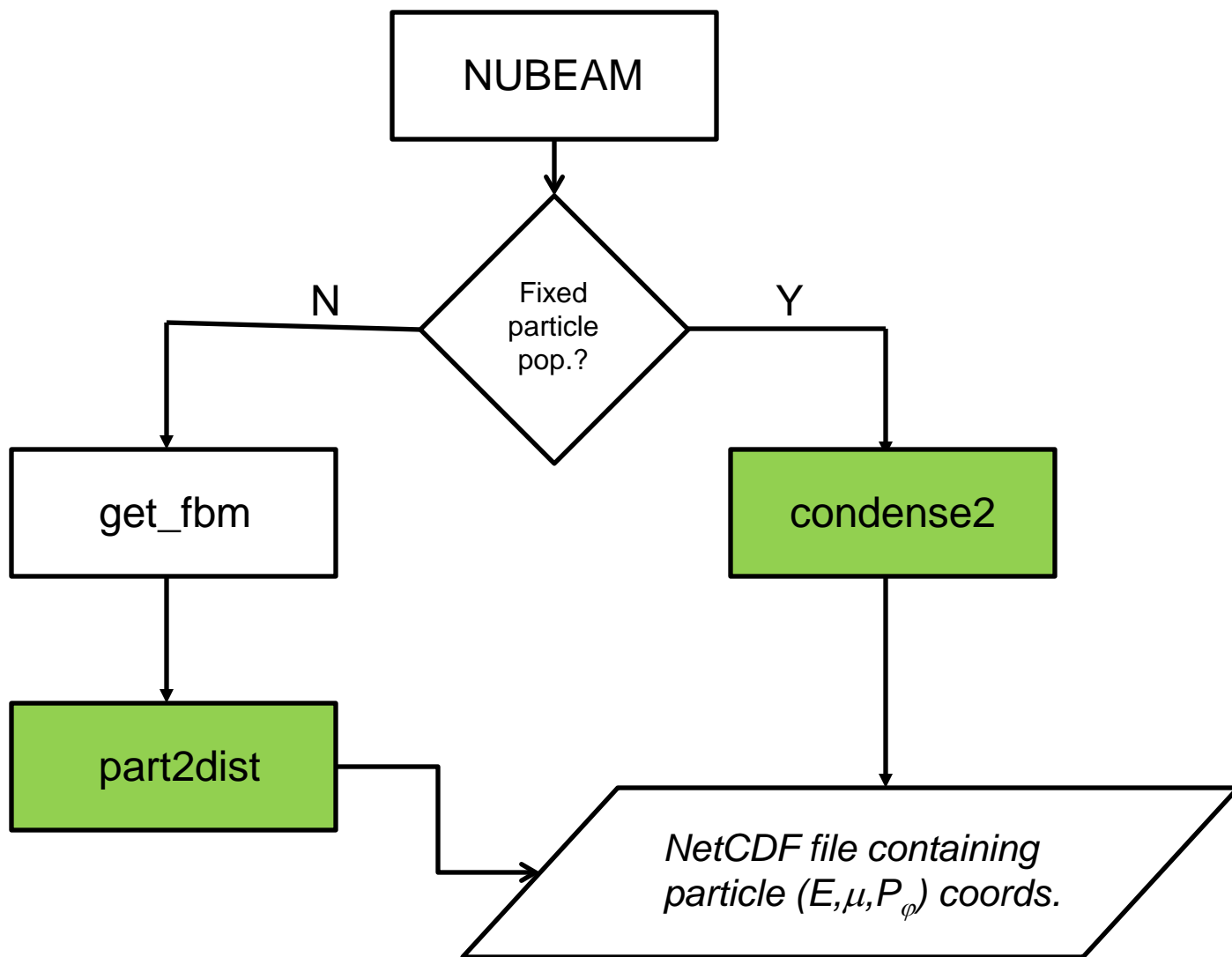


get_fbm: NTCC Utility for sampling 4D distribution function

`% get_fbm @cdf_read.ind` (sample size = 10,000,000)



Workflow, pt. 2: transform coords



Transform to 3D Coordinates

$$F(R, z, \lambda, E) \rightarrow f(P_\varphi, \mu, E)$$

Convert (R,z) to meters, E to Joules.

Then transform using

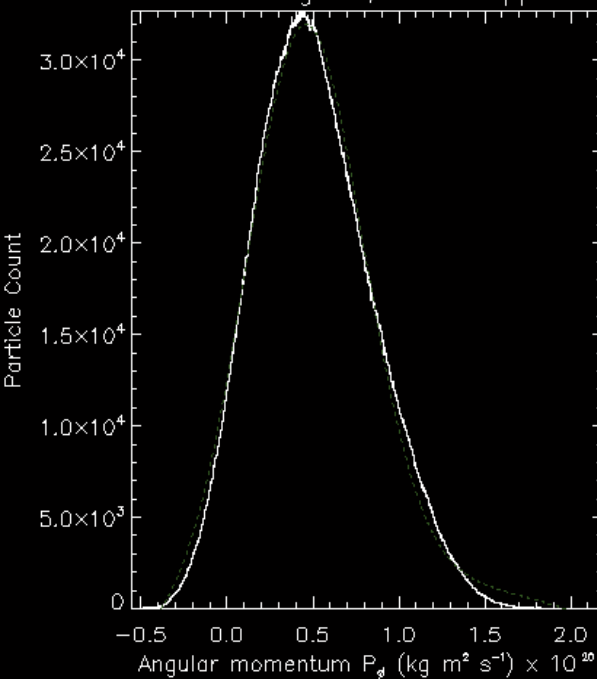
$$P_\varphi \equiv (eA_\varphi + mv_\varphi)R \approx e\psi(R, z) + mv_\parallel R \frac{B_\varphi(R, z)}{B(R, z)}$$

$$\mu = \frac{\frac{1}{2}mv_\perp^2}{B} = \frac{\left[1 - \left(\frac{v_\parallel}{v}\right)^2\right]E}{B(R, z)}$$

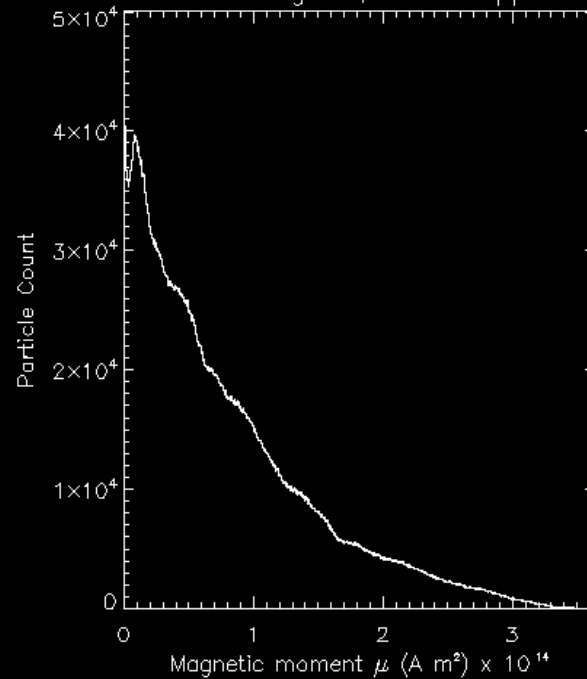
Utility part2dist reads particle data, IPS plasma state, uses plasma state interpolation routines to get ψ , B and perform coordinate transform.

Utility condense2 consolidates relevant particle constants-of-motion coordinates from multi-core run output into a single file.

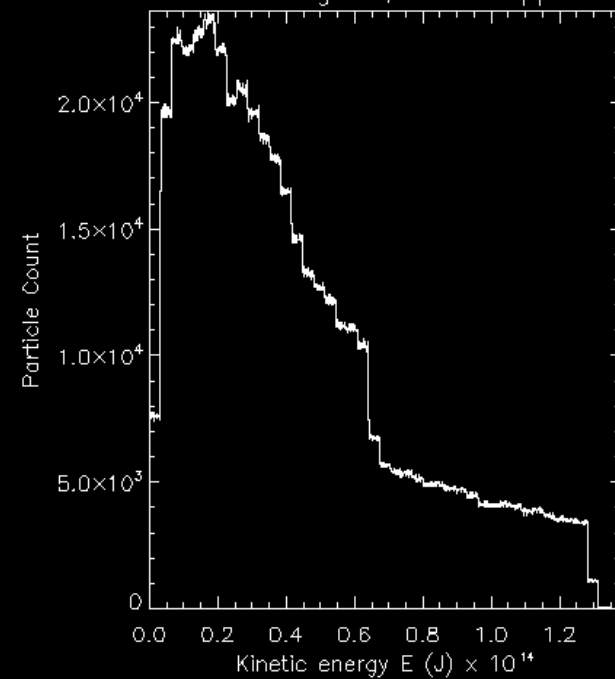
1D Histogram, 10000 ppb



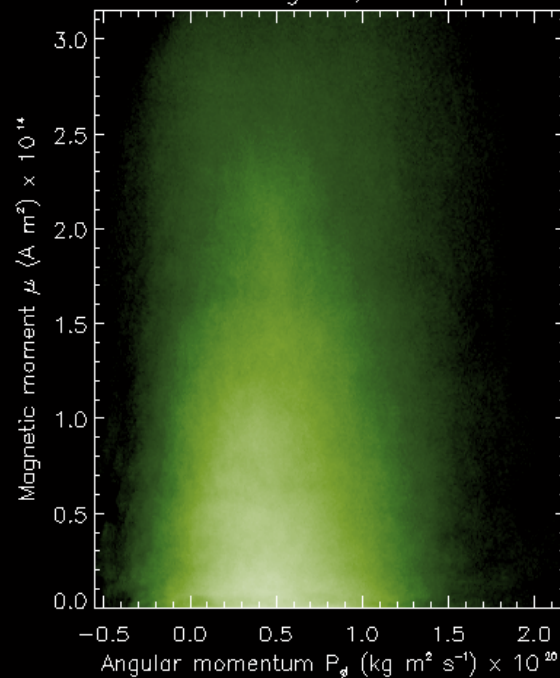
1D Histogram, 10000 ppb



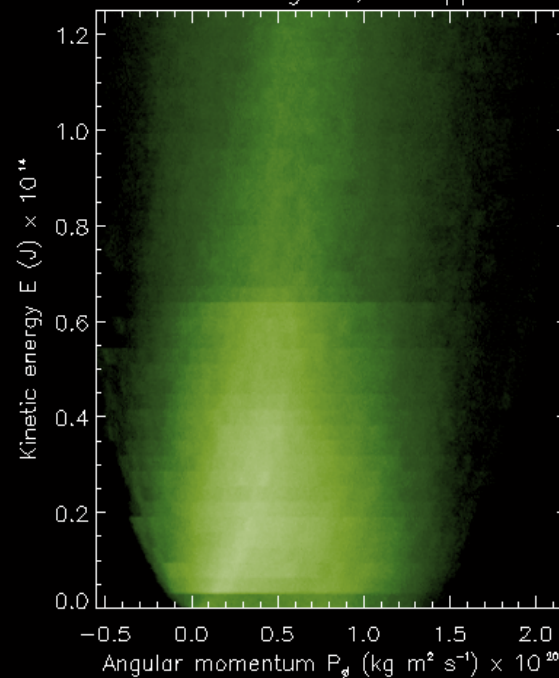
1D Histogram, 10000 ppb



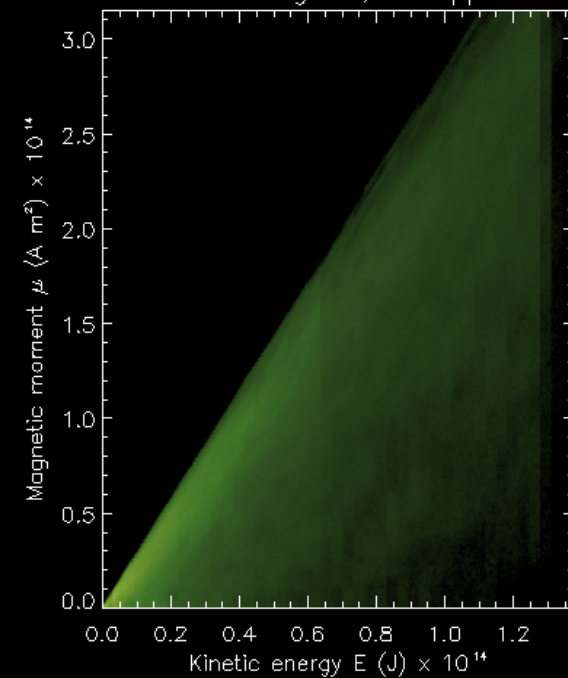
2D Histogram, 100 ppb



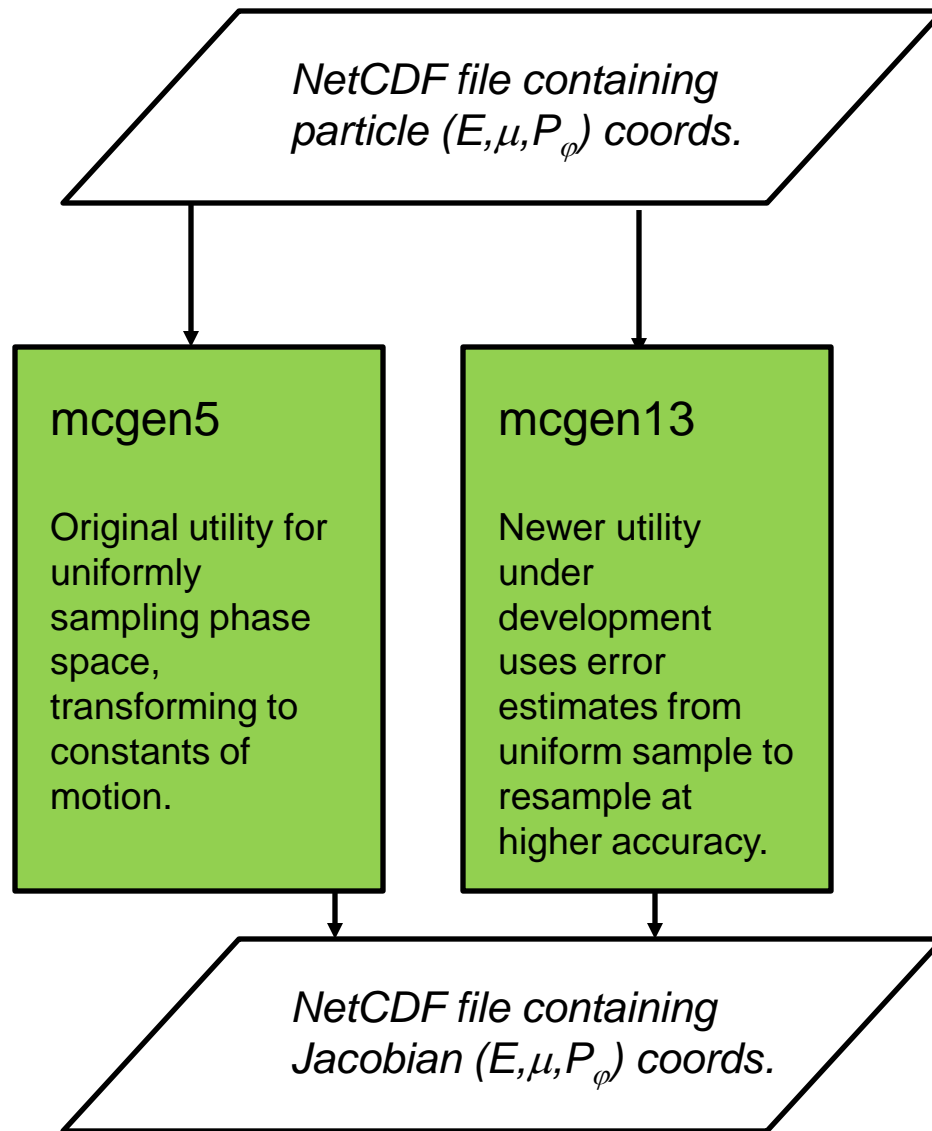
2D Histogram, 100 ppb



2D Histogram, 100 ppb



Workflow, pt. 3: construct Jacobian



Jacobian of the Transform

Over any region of phase space,

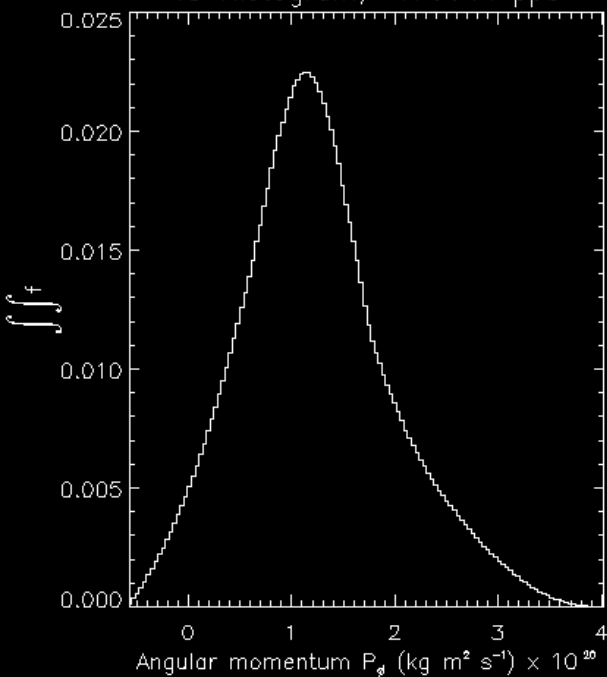
$$\iiint \iiint F(x, y, z, v_x, v_y, v_z) d^3x d^3v = \iiint f(P_\phi, \mu, E) \mathfrak{J}(P_\phi, \mu, E) dP_\phi d\mu dE$$

An exact analytic form for \mathfrak{J} cannot be derived simply as it requires integration over all phase space orbits. A Monte Carlo approach is used, based on the observation that the transform of a uniform distribution in Euclidean phase space is proportional to the inverse of the Jacobian in the new space.

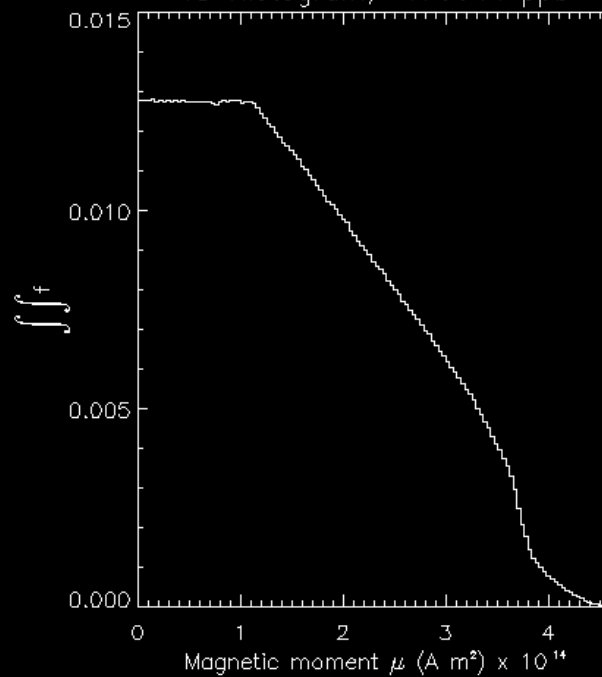
Procedure:

- Generate a population of ~20 million particles in phase space such that R^2 , z , v_\perp^2 , and v_\parallel are uniform random deviates, within the same range as the NUBEAM distribution.
- Transform coordinates to constants-of-motion space.

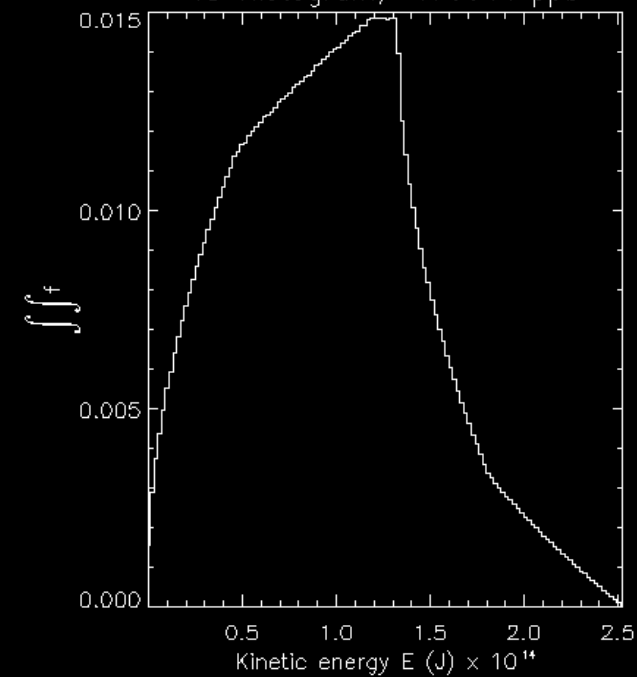
1D Histogram, 179044 ppb



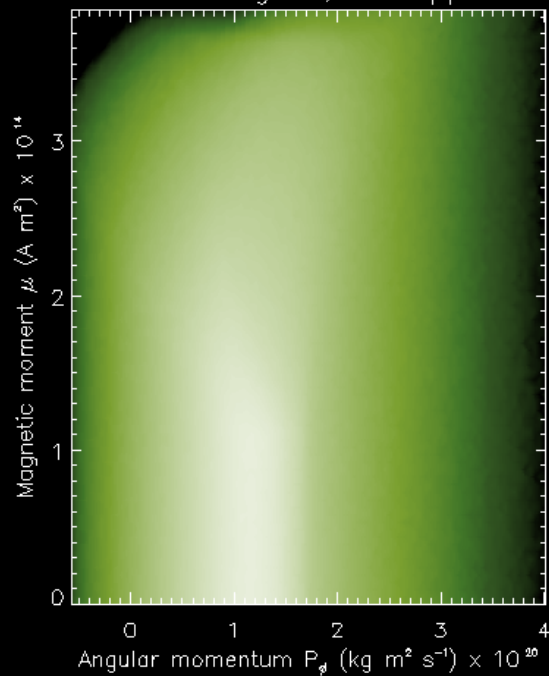
1D Histogram, 179044 ppb



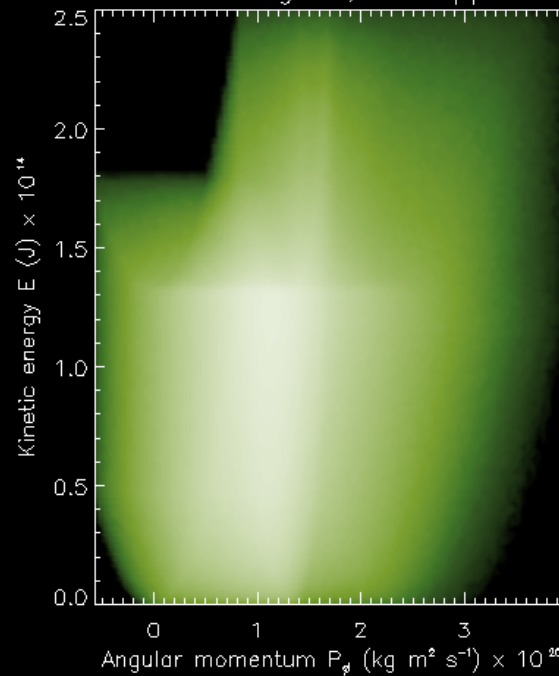
1D Histogram, 179044 ppb



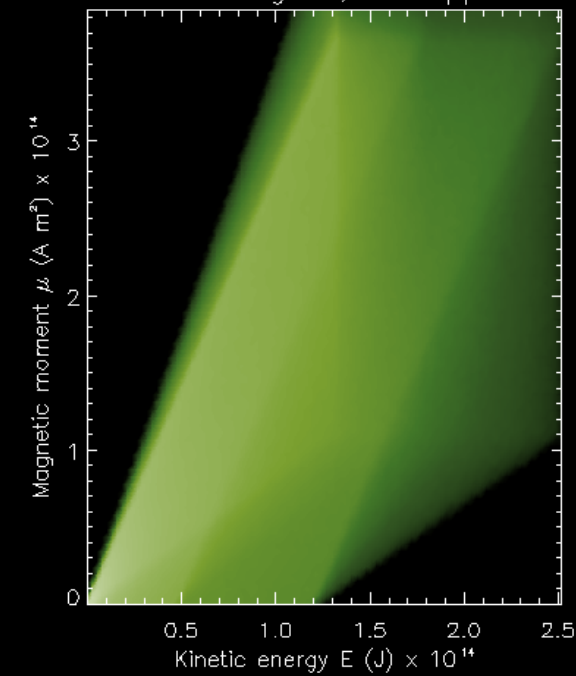
2D Histogram, 2486 ppb



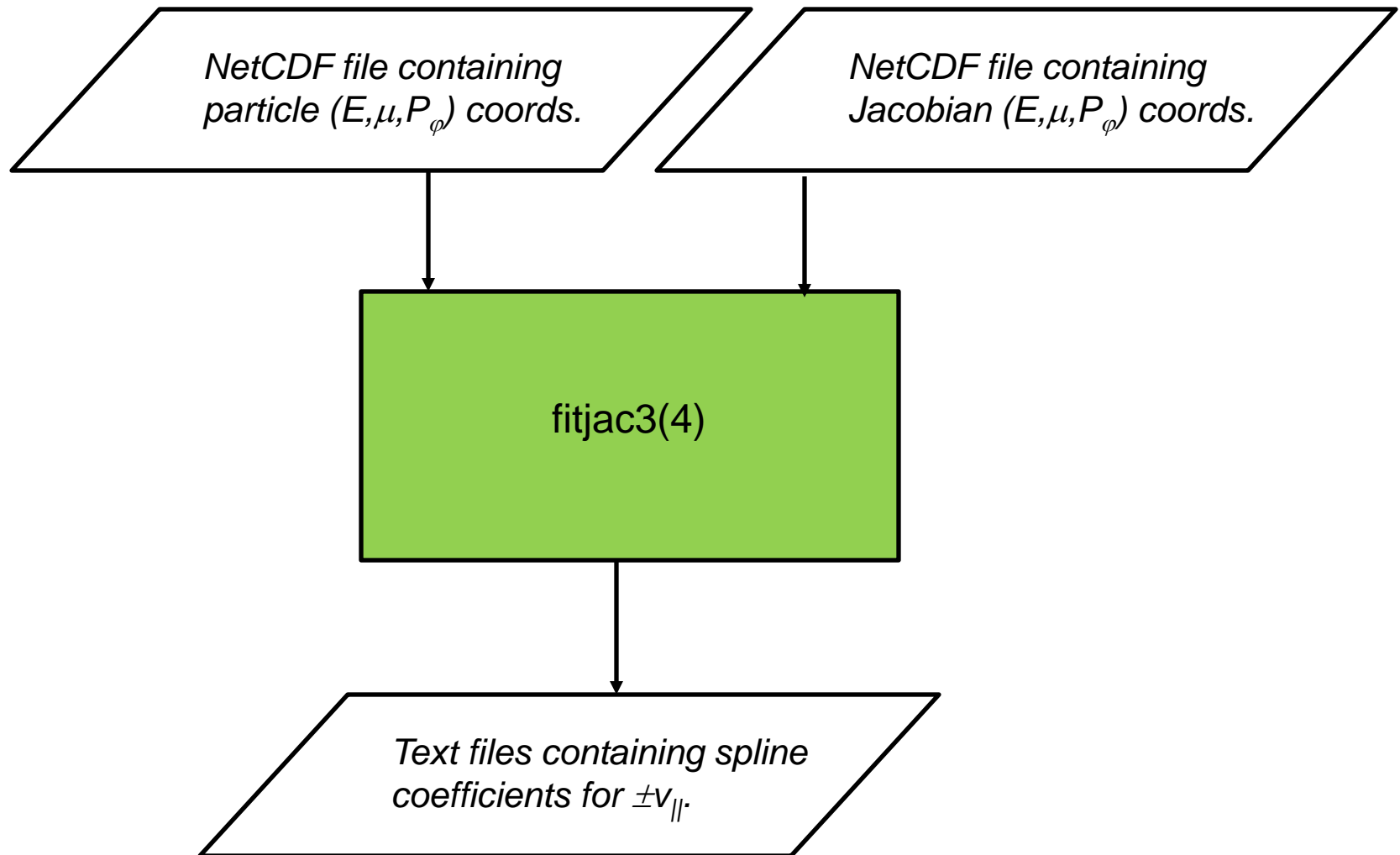
2D Histogram, 2486 ppb



2D Histogram, 2486 ppb







Workflow, pt. 4: Bin, smooth, fit

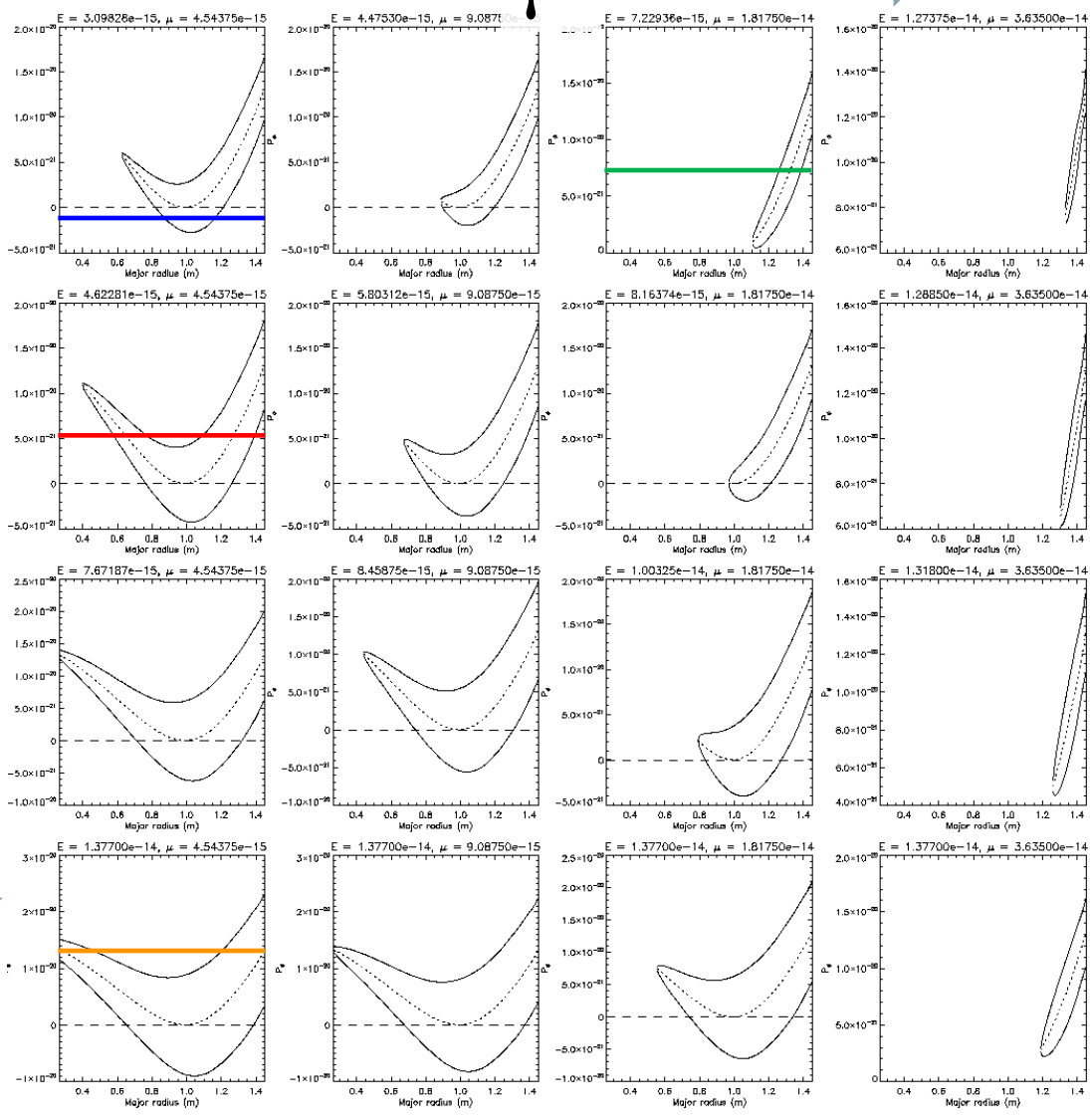


Ambiguity in P_ϕ : $v_{||} = \pm \sqrt{\frac{E - \mu B}{m/2}}$



Cases:

-  Only the negative root is consistent with the constants of motion: particle is co-passing.
-  Only the positive root is consistent with the constants of motion: particle is counter-passing.
-  Positive and negative roots on outboard side are both consistent with the constants of motion: particle is trapped (in a banana orbit).
-  Multiple solutions are consistent with the constants of motion: additional information on the sign of $v_{||}$ is needed to resolve orbit type.



All plots are along midplane.

Constructing $f(P_\phi, \mu, E)$ with `fitjac`

- Divide the particles into subsets according to sign of v_{\parallel} and sort each subset by magnetic moment. Divide them into several subpopulations of either equal width in μ or equal particle count.
- Sort the particles in each subpopulation into a number of bins in the P_ϕ and E directions, chosen heuristically to preserve information but minimize noise.
- Remove “lost” particles, defined as bins containing a single particle with neighbors containing none.
- Apply Gaussian smoothing in both directions.
- Multiply by smoothed, binned Jacobian.
- Fit 2D cubic B-splines to the smoothed data using `gsl` routines, with uniform knots and a number of coefficients in each direction approximately 5/8 the number of bins.
- Store the spline coefficients in a file. Utility routines have been developed to read the spline data file and perform quick spline and derivative interpolations at arbitrary points in the domain. (Linear interpolation is performed in μ).

Reconstruction Routines

Fortran Interface in particleSplines.c:

pspline_init_()

Reads particle spline coefficients from text files, initializes data structures to store them. Returns number of μ bins for each orbit type ($v_{||}$ neg/pos).

getpsplinebounds_()

Returns upper and lower bounds of particle distribution function space in each of the three constants of motion P_φ, μ, E . (For E , $\min|E/\mu|=B_{min}$ sets the lower bound.)

getpdf_($P_\varphi, \mu, E, \text{sgn}, f$)

Returns normalized $f(P_\varphi, \mu, E)$.

getpdfd_($P_\varphi, \mu, E, \text{sgn}, f, dfdp, dfdE$)

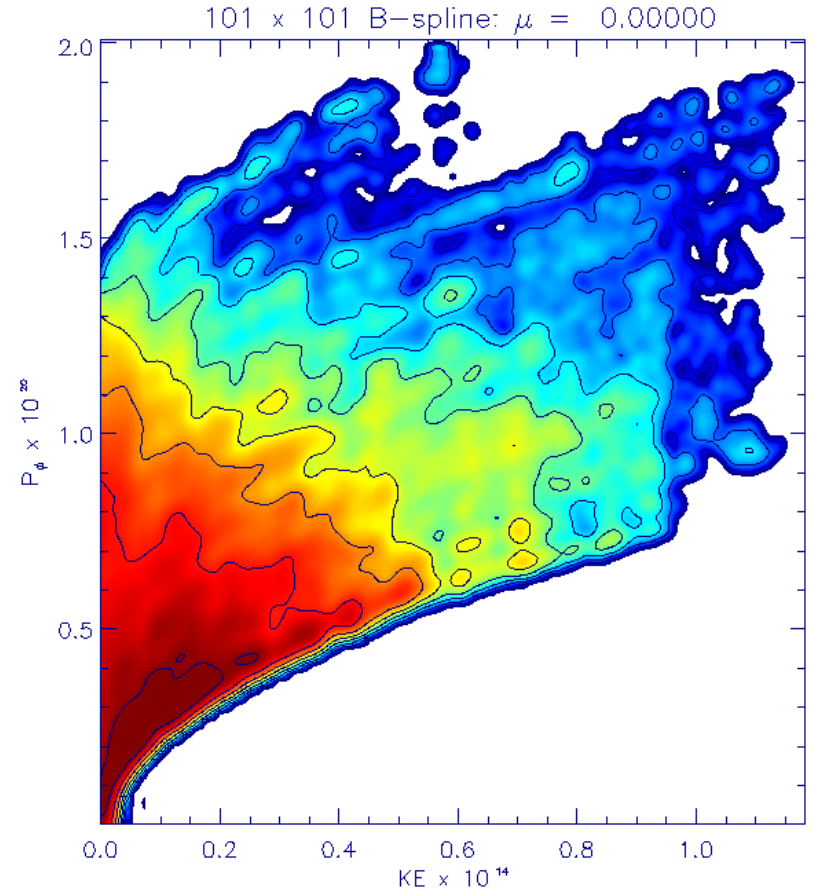
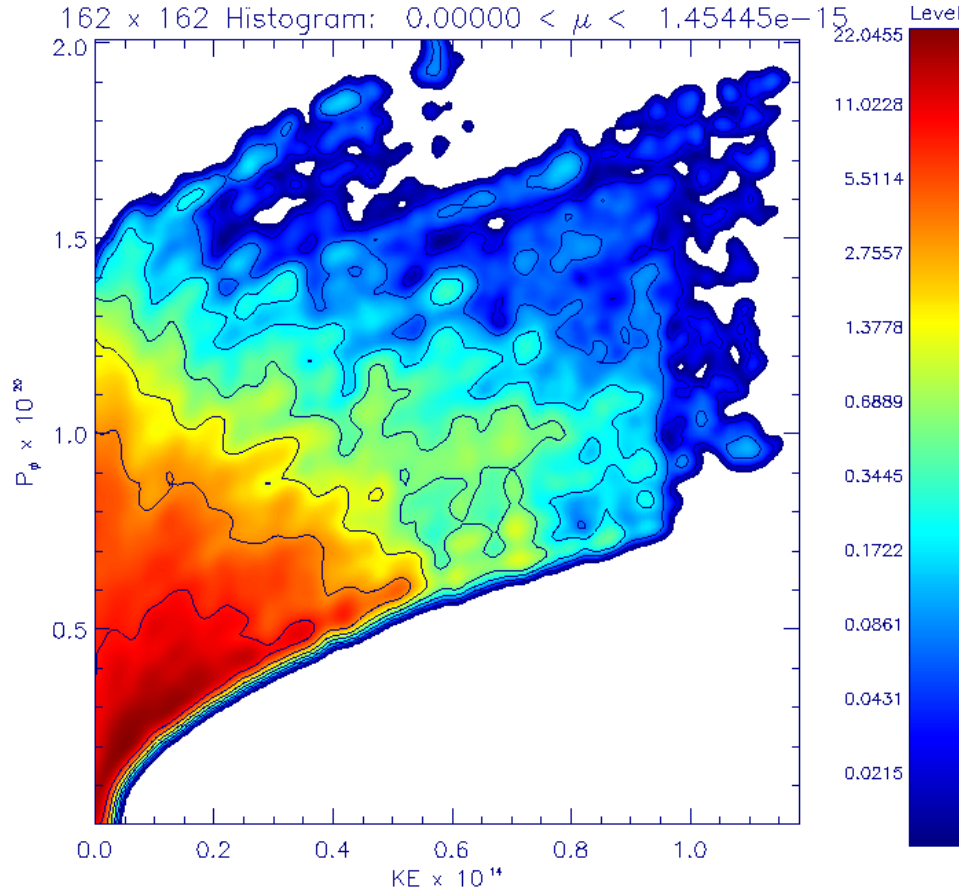
Returns normalized $f(P_\varphi, \mu, E)$, $\partial f/\partial P_\varphi$, and $\partial f/\partial E$.

pspline_free_()

Frees up all storage associated with particle spline data structures.

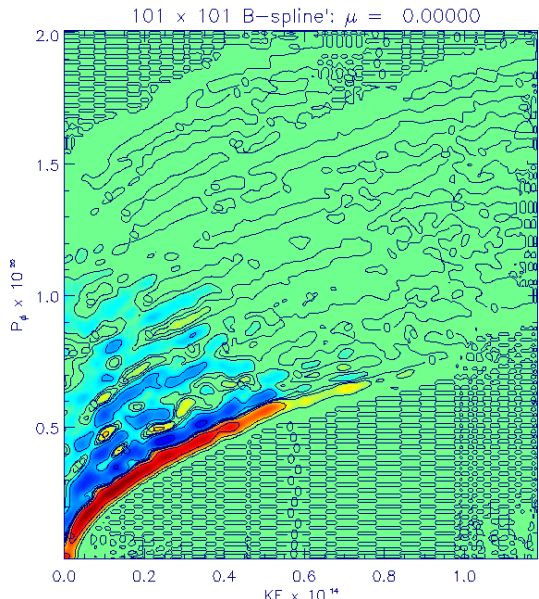
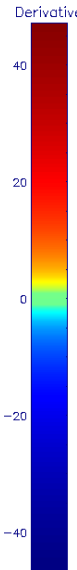
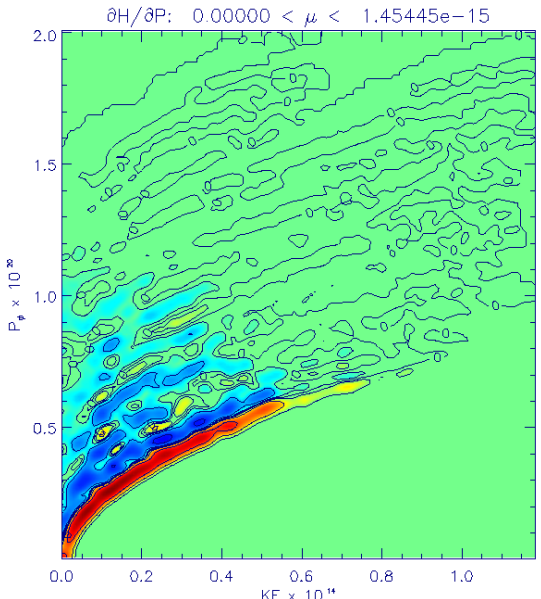
High-Occupancy Bin Fits Are Good

$\lambda > 0$ Bin 1/25: 1,043,794 particles

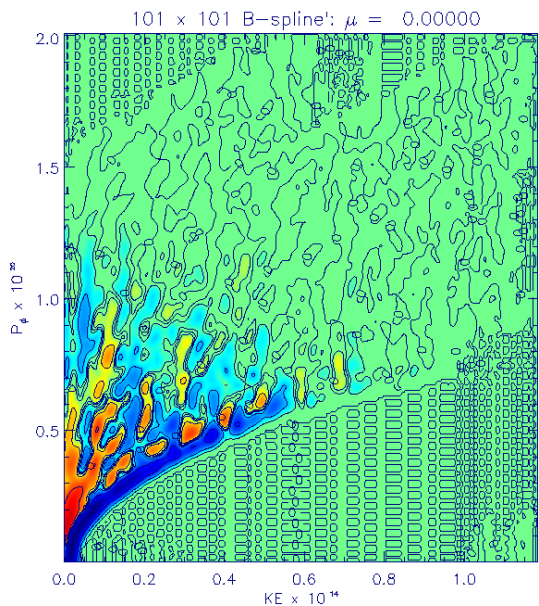
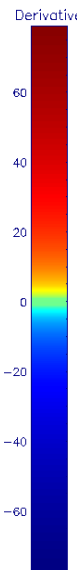
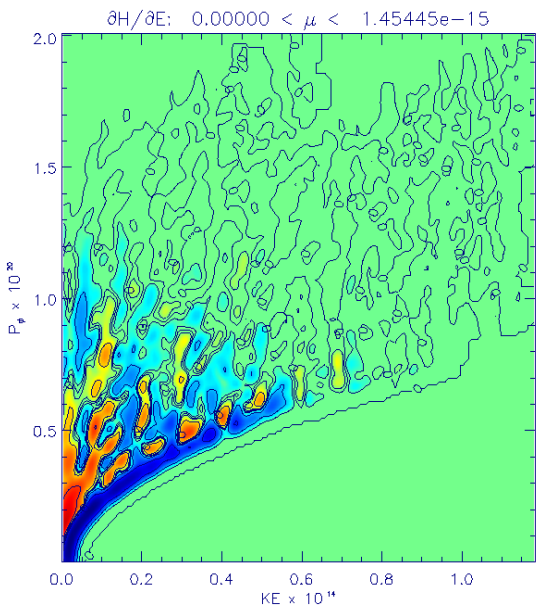


$$\sigma_P = \sigma_E = 1.334$$

Gradients match as well



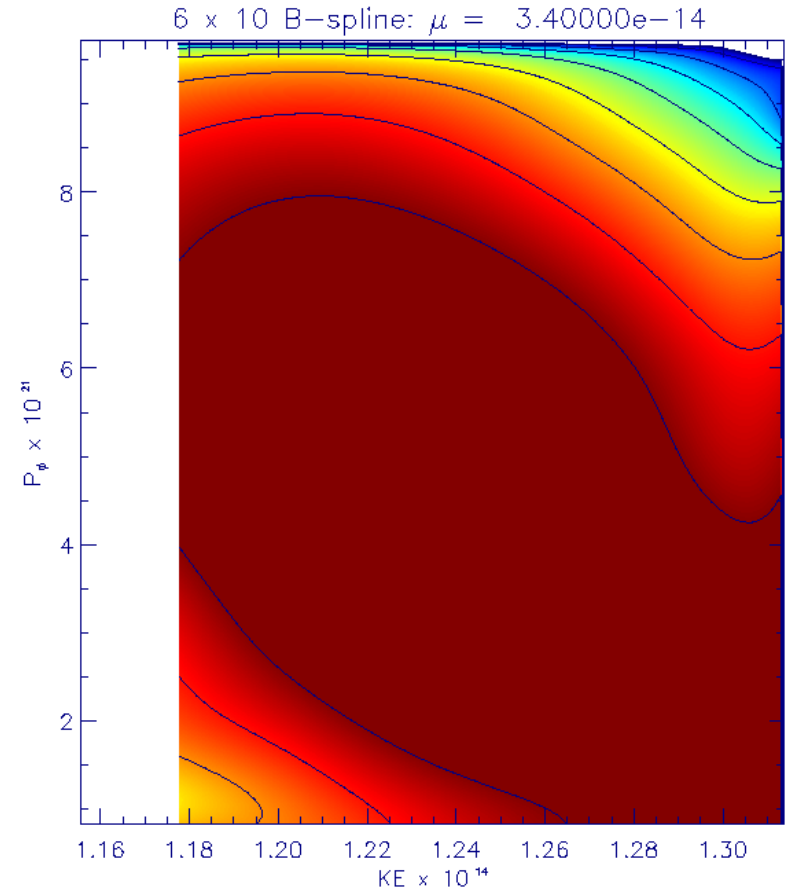
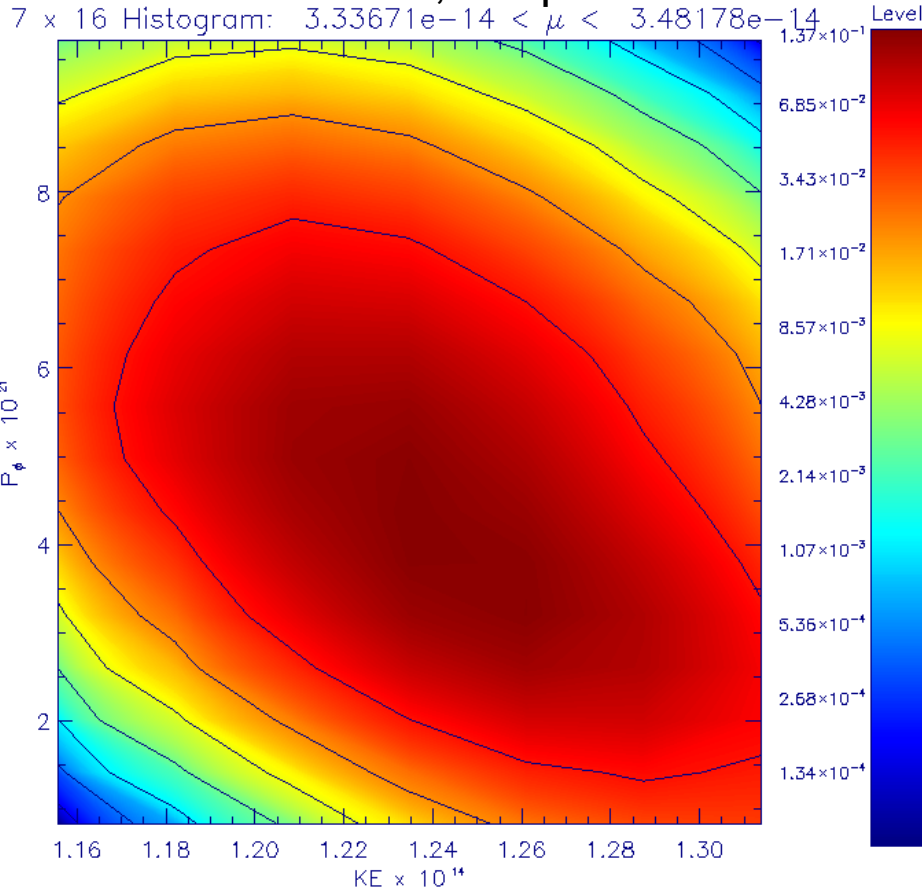
$$\frac{\partial f^+}{\partial P_\phi}$$



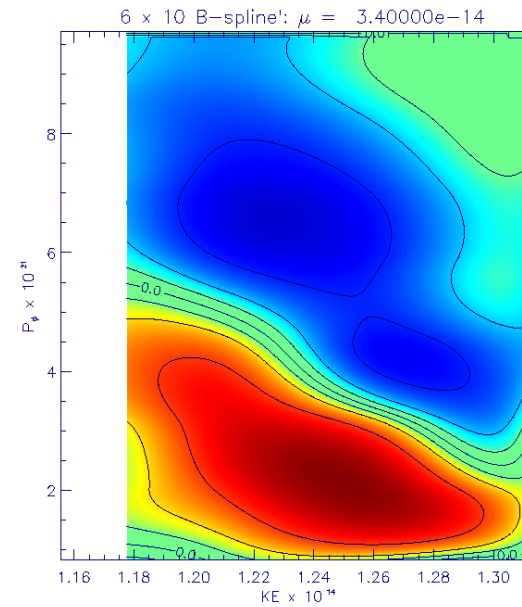
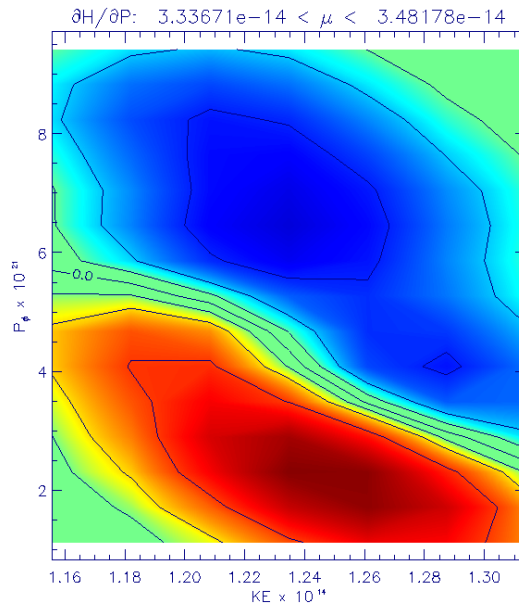
$$\frac{\partial f^+}{\partial E}$$

Low-Occupancy Fits are Adequate

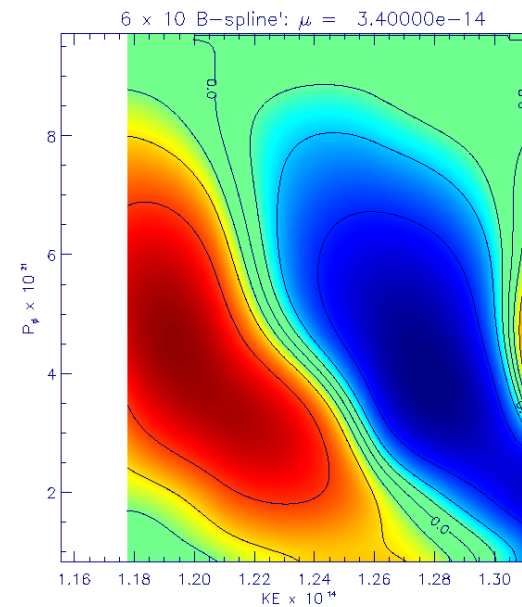
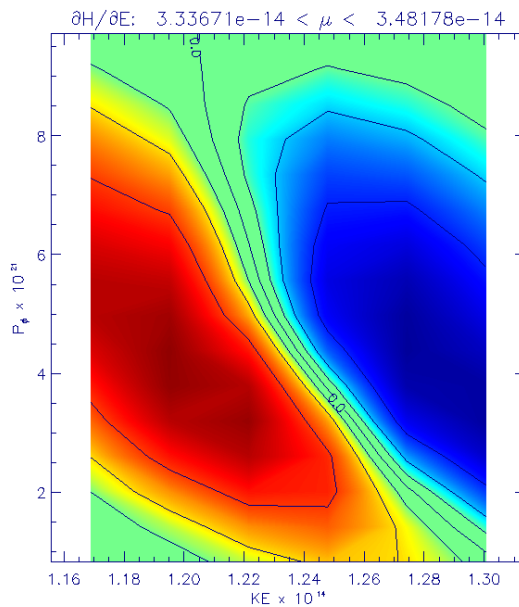
$\lambda < 0$ Bin 24/25: 1,208 particles



Gradients are Smoothed



$$\frac{\partial f^-}{\partial P_\phi}$$



$$\frac{\partial f^-}{\partial E}$$

Project Status

- All utilities have been gathered to a single location, can be built with a single Makefile.

 - Includes condense2.c, mcgen5.f90, fitjac3.c, spline_interface.c, particleSplines.c, testEsplines5.f90 (sample usage).

 - Requires modules ntcc/tshare, netcdf, mdsplus, gsl.

- mcgen13 requires further optimization.

- Post-processing workflow for generation of spline files from NUBEAM data has been simplified into a single, simple Python script.

 - Will add more knobs for users.

 - Needs more testing to verify accuracy and to optimize sampling, fit parameters.

- Fortran interface for spline reconstruction of profiles may be adapted based on user feedback.