q	float	Charge on electron (magnitude)
kB	float	Boltzmann’s constant
eps0	float	Permittivity of free space
epsr	float	Dielectric constant
NT	float	Trap density
T	float	Temperature
refin	float	Refractive index
noiseamp	float	Scattering noise amplitude
Id	float	Dark intensity
sigma	float	Noise correlation length
gl	float	Coupling constant
xaper	float	Geometry aperture in x direction
yaper	float	Geometry aperture in y direction
xsamp	int	Number of sampling points in x
ysamp	int	Number of sampling points in y
rlen	float	Medium length, all coords in microns
dz	float	Step size
dz_frac	float	Delta_z/dz
windowedge	float	Tukey window edge width parameter
geom	tuple; xaper,yaper,xsamp,ysamp,rlen,dz	Tuple to check if geometry has changed from last instance used
lm	float	Wavelength microns
w01	float	Beam 1 waist
w02	float	Beam 2 waist
thout1	float	Beam 1 external angle of incidence
thout2	float	Beam 2 external angle of incidence
phi1	float	Beam 1 azimuth
phi2	float	Beam 2 azimuth
rat	float	Beam ratio I2/I1
mode	String #’fdbpm’ or ‘fft’ or ‘fdbpm nonparaxial’ or ‘Gaussian Analytical’	Computation mode
transp1	boolean	Switch when true applies image to beam 1
transp2	boolean	Switch when true applies image to beam 2
loc	String PxPy or PyQx	Switch to set P or Q mode in FDBPM
storenoise	boolean	Switch if true causes 3D noise to be stored in prdata, resulting in large increase in file size
data_in_size	float; w01	Size of image to impress on beam
corrnoise	<class 'numpy.ndarray'> float64 (rlen/dz, xsamp, ysamp);rlen,dz	3D noise array if needed
h	<class 'numpy.ndarray'> complex128 (xsamp, ysamp); lm,refin,fxy <-fx,fy<-xsamp,ysamp,dz,	fft propagator
xp1	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th1<-thout1,refin	Beam 1 gaussian xp in terms of grid x and y coord
yp1	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th1<-thout1,refin	Beam 1 gaussian yp in terms of grid x and y coord

zp1	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th1<-thout1,refin	Beam 1 gaussian zp in terms of grid x and y coord
xp1dz	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th1<-thout1,refin,dz_small	Beam 1 gaussian xp in terms of grid x and y coord at dz_small from input face
yp1dz	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th1<-thout1,refin,dz_small	Beam 1 gaussian yp in terms of grid x and y coord at dz_small from input face
zp1dz	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th1<-thout1,refin,dz_small	Beam 1 gaussian zp in terms of grid x and y coord at dz_small from input face
xp2	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th2<-thout2,refin	Beam 2 gaussian xp in terms of grid x and y coord
yp2	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th2<-thout2,refin	Beam 2 gaussian yp in terms of grid x and y coord
zp2	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th2<-thout2,refin	Beam 2 gaussian zp in terms of grid x and y coord
xp2dz	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th2<-thout2,refin,dz_small	Beam 2 gaussian xp in terms of grid x and y coord at dz_small from input face
yp2dz	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th2<-thout2,refin,dz_small	Beam 2 gaussian yp in terms of grid x and y coord at dz_small from input face
zp2dz	<class 'numpy.ndarray'> complex128 (xsamp, ysamp);phi1,th2<-thout2,refin,dz_small	Beam 2 gaussian zp in terms of grid x and y coord at dz_small from input face
A	<xsamp x xsamp sparse matrix of type '<class 'numpy.float64'>', xsamp,dx,<-xaper,xsamp with **** stored elements in Compressed Sparse Row format>	tranverse x laplacian for Q form of BPM operator
B	<ysamp x ysamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format> ysamp,dy,<-yaper,ysamp	transverse y laplacian for Q form of BPM operator
Px	[[<xsamp x xsamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>, <xsamp x xsamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>], [<xsampx x samp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>, <xsamp x xsamp sparse matrix of type '<class 'numpy.float64'>'	transverse x propagator for P form of BPM operator

	with xxxx stored elements in Compressed Sparse Row format>]], dz,xsamp,kin<-refin,lm	
Py	[[<ysamp x ysamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>, <ysamp x ysamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>], [<ysamp x ysamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>, <ysamp x ysamp sparse matrix of type '<class 'numpy.float64'>' with xxxx stored elements in Compressed Sparse Row format>]], dz,ysamp,kin<-refin,lm	transverse y propagator for P form of BPM operator

Prdata dictionary entries for beam propagation code