Read .ems images using Mathematica

P. Stadelmann
EPFL - CIME
CH - 1015 Lausanne
Pierre.Stadelmann@epfl.ch

Setting data path

Notebook is assumed to be in the images folder

```
SetDirectory [ NotebookDirectory []]
```

```
/Users/pierrestadelmann/Desktop/Zut
```

Read real images

image is scaled from [min, max] are mapped on [0, 1].

```
readRealImage [name_String] := Module [{col, imager, is, row, maxr, minr, xr},
  is = OpenRead[name, BinaryFormat → True];
  row = BinaryRead[is, "Integer32", ByteOrdering → 1];
  col = BinaryRead [is, "Integer32", ByteOrdering → 1];
  xr = BinaryRead [is, Table ["Real32", {col * row}], ByteOrdering → 1];
  xr = xr // Partition [#, row] &;
  Close [is];
  maxr = Max [xr];
  minr = Min [xr];
  imager = Image [ Raster [(xr - minr) / (maxr - minr)]];
  imager
 ]
```

Read complex or Fourier images

Image real and imaginary part from [min, max] are mapped on [0, 1].

```
readComplexImage [name_String] :=
 Module [{col, comp, imagei, imager, is, row, maxr, minr, maxi, mini, xr, xi},
  is = OpenRead[name, BinaryFormat → True];
  row = BinaryRead[is, "Integer32", ByteOrdering → 1];
  col = BinaryRead [is, "Integer32", ByteOrdering → 1];
  xr = BinaryRead [is, Table ["Real32", {col * row}], ByteOrdering → 1];
  xr = xr // Partition [#, row] &;
  comp = BinaryRead [is, "Integer32", ByteOrdering → 1];
  xi = BinaryRead [is, Table ["Real32", {col * row}], ByteOrdering → 1];
  xi = xi // Partition [#, row] &;
  Close [is];
  maxr = Max [xr];
  minr = Min [xr];
  imager = Image [ Raster [(xr - minr) / (maxr - minr)]];
  maxi = Max [xi];
  mini = Min [xi];
  imagei = Image [ Raster [(xi - mini) / (maxi - mini)]];
  {imager, imagei}
 ]
```

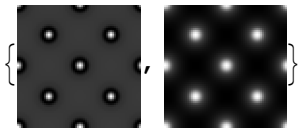
Multislice wavefunction

```
msWaves = FileNames ["y*"]
```

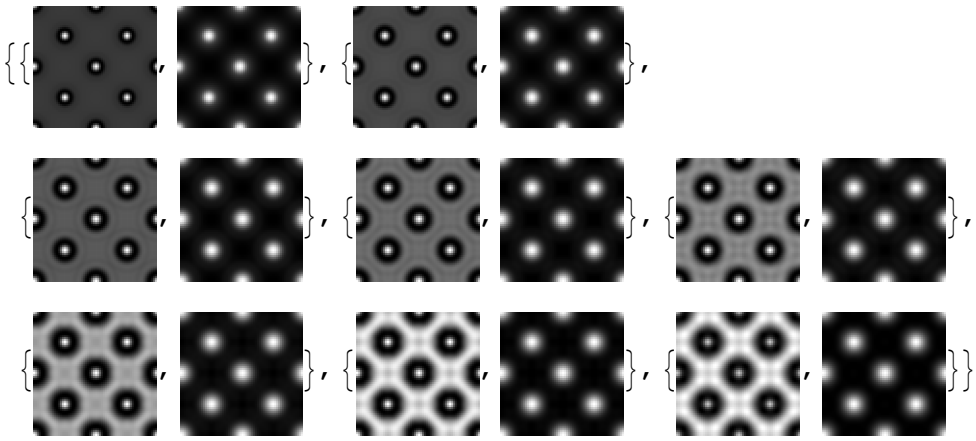
```
{y0.ems, y1.ems, y2.ems, y3.ems, y4.ems, y5.ems, y6.ems, y7.ems}
```

y* images calculated using Multislice method. First wavefunction at z >0

```
readComplexImage [msWaves[[1]]]
```



```
readComplexImage [msWaves[[#]]] & /@ Range@Length@msWaves
```



Blochwave wavefunction

```
bwWaves = FileNames ["z_*.ems"]
```

```
{z_0000.ems, z_0001.ems, z_0002.ems, z_0003.ems, z_0004.ems,
 z_0005.ems, z_0006.ems, z_0007.ems, z_0008.ems, z_0009.ems,
 z_0010.ems, z_0011.ems, z_0012.ems, z_0013.ems, z_0014.ems, z_0015.ems,
 z_0016.ems, z_0017.ems, z_0018.ems, z_0019.ems, z_0020.ems, z_0021.ems,
 z_0022.ems, z_0023.ems, z_0024.ems, z_0025.ems, z_0026.ems, z_0027.ems,
 z_0028.ems, z_0029.ems, z_0030.ems, z_0031.ems, z_0032.ems, z_0033.ems,
 z_0034.ems, z_0035.ems, z_0036.ems, z_0037.ems, z_0038.ems, z_0039.ems,
 z_0040.ems, z_0041.ems, z_0042.ems, z_0043.ems, z_0044.ems, z_0045.ems,
 z_0046.ems, z_0047.ems, z_0048.ems, z_0049.ems, z_0050.ems, z_0051.ems,
 z_0052.ems, z_0053.ems, z_0054.ems, z_0055.ems, z_0056.ems, z_0057.ems,
 z_0058.ems, z_0059.ems, z_0060.ems, z_0061.ems, z_0062.ems, z_0063.ems,
 z_0064.ems, z_0065.ems, z_0066.ems, z_0067.ems, z_0068.ems, z_0069.ems,
 z_0070.ems, z_0071.ems, z_0072.ems, z_0073.ems, z_0074.ems, z_0075.ems,
 z_0076.ems, z_0077.ems, z_0078.ems, z_0079.ems, z_0080.ems, z_0081.ems,
 z_0082.ems, z_0083.ems, z_0084.ems, z_0085.ems, z_0086.ems, z_0087.ems,
 z_0088.ems, z_0089.ems, z_0090.ems, z_0091.ems, z_0092.ems, z_0093.ems,
 z_0094.ems, z_0095.ems, z_0096.ems, z_0097.ems, z_0098.ems, z_0099.ems}
```

First BW wavefunction calculated a z = 0. Can' t be read or used by the Imager.

```
readComplexImage [bwWaves ⟦1⟧]
```

Power::infy : Infinite expression $\frac{1}{0.}$ encountered. ≫

Infinity::indet : Indeterminate expression 0. ComplexInfinity encountered. ≫

Infinity::indet : Indeterminate expression 0. ComplexInfinity encountered. ≫

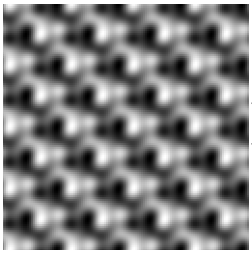
Infinity::indet : Indeterminate expression 0. ComplexInfinity encountered. ≫

General::stop : Further output of Infinity::indet will be suppressed during this calculation. ≫

Image::imgarray : The specified argument
Raster[{{Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, ≪32≫, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate, ≪78≫}, ≪49≫, ≪78≫}] should be an array of rank 2 or 3 with machine-sized numbers. ≫

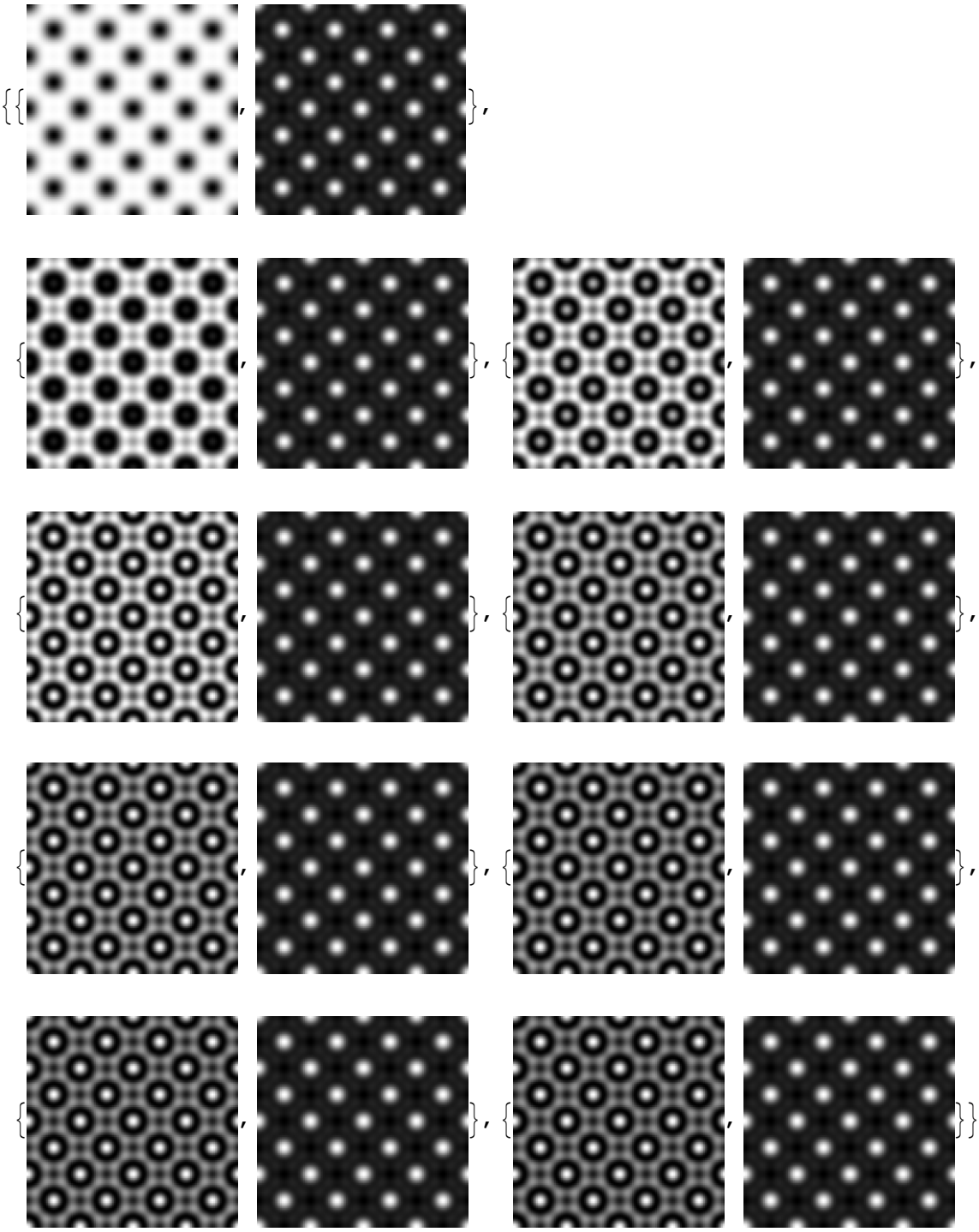
A very large output was generated. Here is a sample of it:

```
{Image[Raster[{{Indeterminate, Indeterminate, Indeterminate, Indeterminate,
    Indeterminate, Indeterminate, Indeterminate, Indeterminate,
    Indeterminate, ≪110≫, Indeterminate, Indeterminate, Indeterminate,
    Indeterminate, Indeterminate, Indeterminate, Indeterminate,
    Indeterminate, Indeterminate}, ≪127≫}]],
```



Show Less | Show More | Show Full Output | Set Size Limit...

```
readComplexImage [bwWaves [[#]] & /@ Range [2, 10]
```

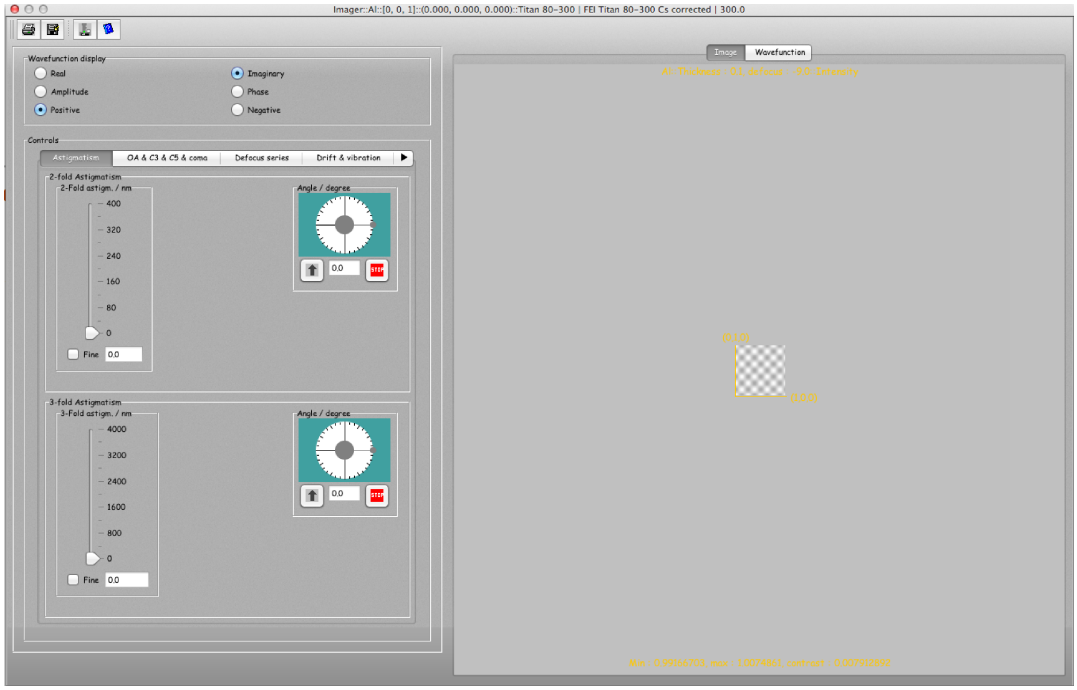


Conclusion

BW

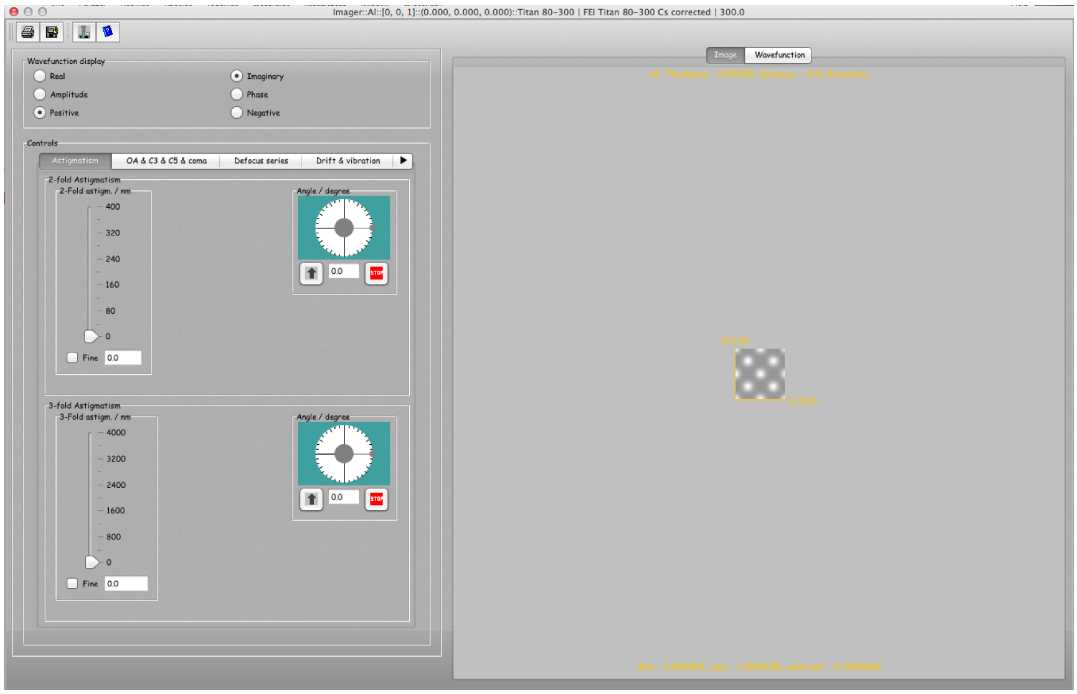
You can use the imager but with BW wavefunction for z > 0.

```
Import ["bwImager.png"]
```



MS

```
Import ["msImager.png"]
```



I' ll have to correct the numbering of the MS wavefunction.