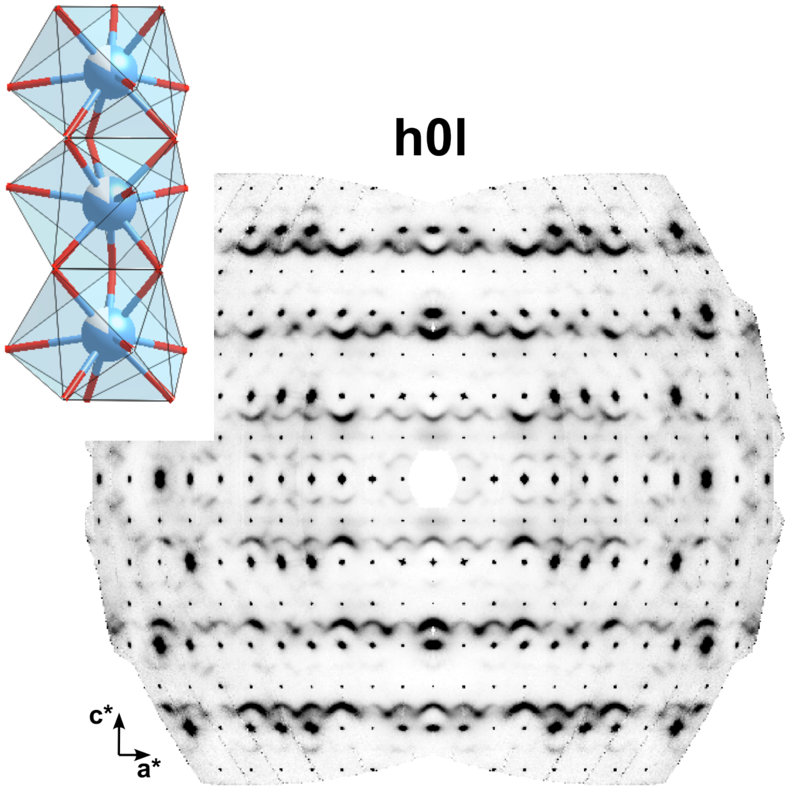
Fast simulation of disordered crystals using Gaussian copula

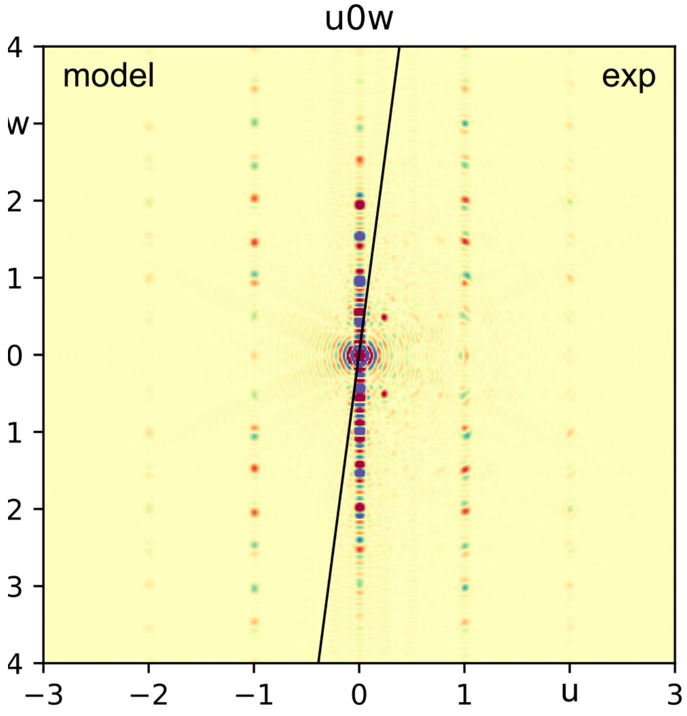
Arkadiy Simonov

ETH Zurich

Crystal structure solution with 3D-ΔPDF



FT

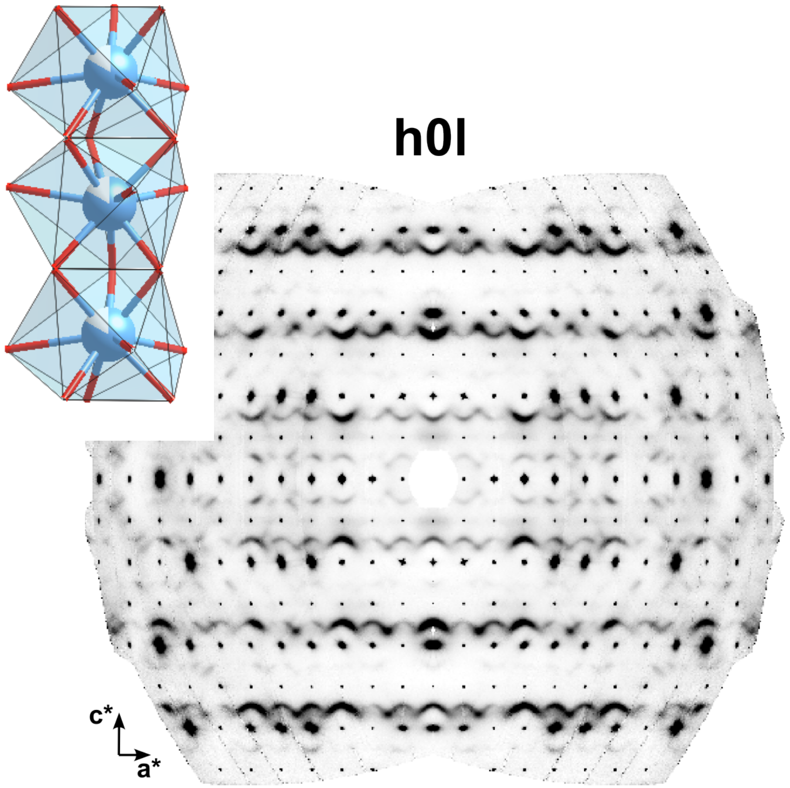


Refined Covariances:

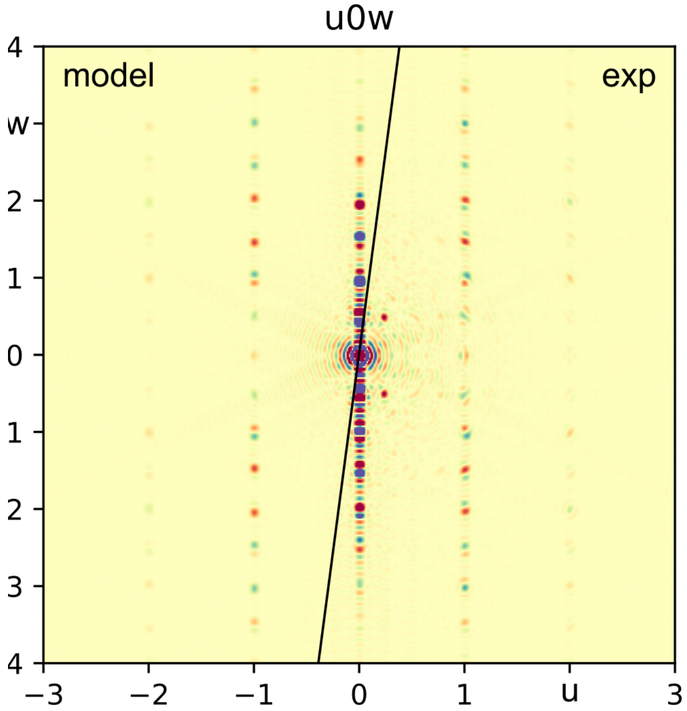
Dy_Dy_dp1=0.01572(3);
Dy_Dy_dp2=0.01971(2);
Dy_Dy_dp3=-0.00102(12);
Dy_Dy_dp4=-0.04645(2);
Dy_Dy_dp5=-0.01466(4);
…

Is model self consistent?
How would a crystal
like this look like?

Crystal structure solution with 3D-ΔPDF



FT



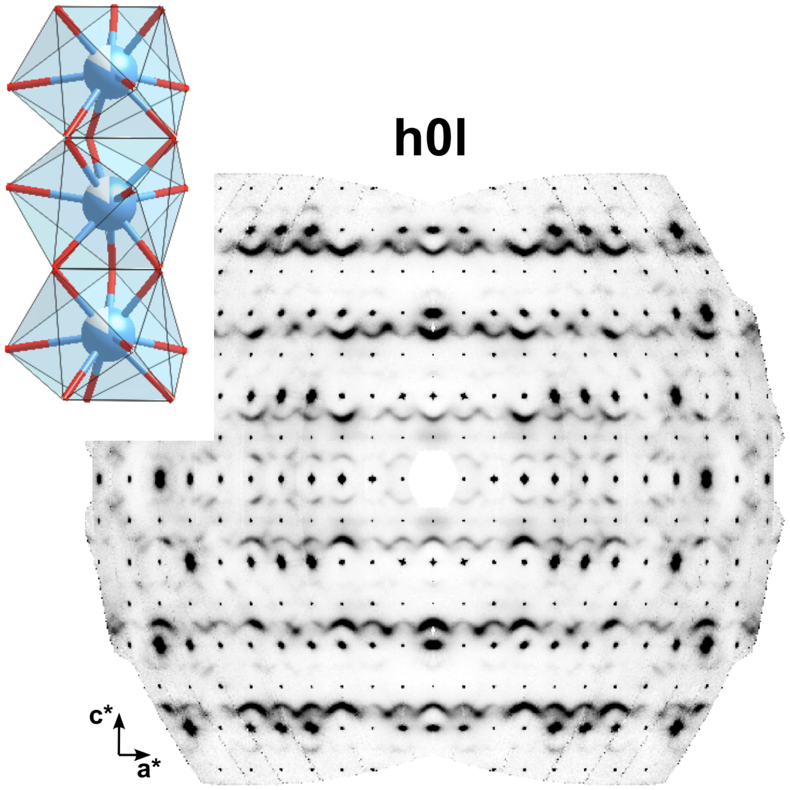
Refined Covariances:

Dy_Dy_dp1=0.01572(3);
Dy_Dy_dp2=0.01971(2);
Dy_Dy_dp3=-0.00102(12);
Dy_Dy_dp4=-0.04645(2);
Dy_Dy_dp5=-0.01466(4);
…

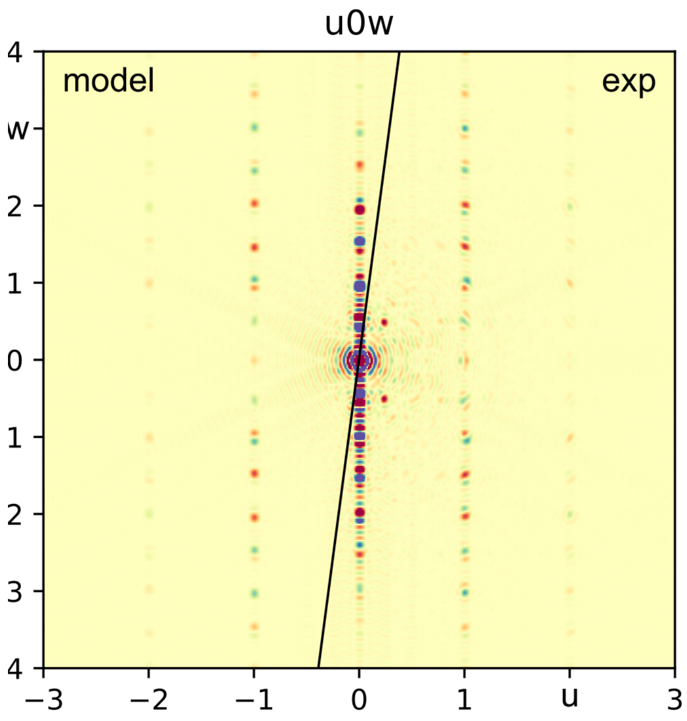
Monte-Carlo

Reverse Monte-Carlo

Crystal structure solution with 3D-ΔPDF



FT



Refined Covariances:

Dy_Dy_dp1=0.01572(3);
Dy_Dy_dp2=0.01971(2);
Dy_Dy_dp3=-0.00102(12);
Dy_Dy_dp4=-0.04645(2);
Dy_Dy_dp5=-0.01466(4);
…

Monte-Carlo

Reverse Monte-Carlo

Gaussian Copula

Mathematical formulation

Generate a disordered crystal

Generate a field of Bernoulli variables
$x_i = \{0,1\}$

… consistent with diffraction

… such that:

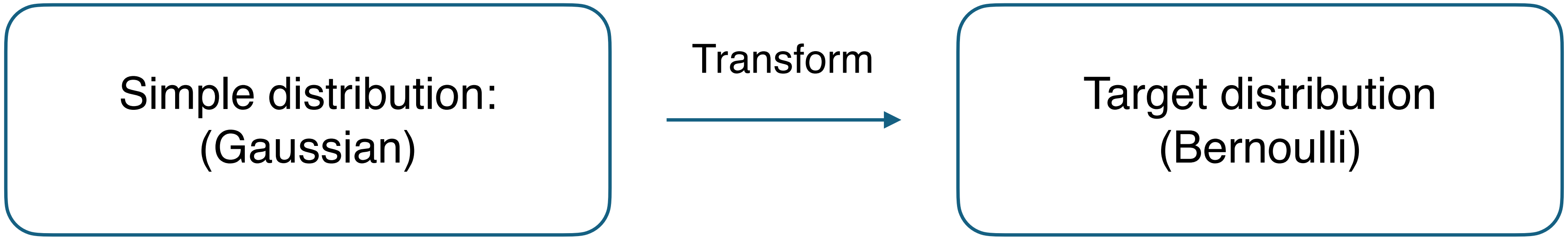
$$\langle x_i \rangle = o_i$$

where o_i is occupancy of site i. For notational simplicity we will assume $o_i = 0.5$

and

$$Cov(x_i, x_j) = C^x_{ij}$$

Idea of distribution modeling via a copula



Gaussian copula generation

$\sim$ Bern$(0.5, \mathbf{C}^x)$

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

[Qing. Comms. Stat Theory and Meth. 46.4 (2017): 1594-1605.]

Gaussian copula generation

$$\sim \mathcal{N}(\mathbf{0}, \mathbf{C}^z) \qquad \sim \mathrm{Bern}(0.5, \mathbf{C}^x)$$

$$\begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} \xrightarrow{x_i = \begin{cases} 1 & \text{if } z_i > 0 \\ 0 & \text{if } z_i \leq 0 \end{cases}} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

[Qing. Comms. Stat Theory and Meth. 46.4 (2017): 1594-1605.]

Gaussian copula generation

$$\sim \mathcal{N}(\mathbf{0}, \mathbf{1}) \qquad \sim \mathcal{N}(\mathbf{0}, \mathbf{C}^z) \qquad \sim \text{Bern}(0.5, \mathbf{C}^x)$$

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \xrightarrow[Z = LU]{C^z = LL^T} \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} \xrightarrow{x_i = \begin{cases} 1 & \text{if } z_i > 0 \\ 0 & \text{if } z_i \leq 0 \end{cases}} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

[Qing. Comms. Stat Theory and Meth. 46.4 (2017): 1594-1605.]

Gaussian copula generation

$$\sim \mathcal{N}(\mathbf{0}, \mathbf{1}) \qquad\qquad \sim \mathcal{N}(\mathbf{0}, \mathbf{C}^z) \qquad\qquad \sim \text{Bern}(0.5, \mathbf{C}^x)$$

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \xrightarrow[]{\substack{C^z = LL^T \\ Z = LU}} \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} \xrightarrow[]{x_i = \begin{cases} 1 & \text{if } z_i > 0 \\ 0 & \text{if } z_i \leq 0 \end{cases}} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Things left to do: find relation between $\mathbf{C}^z$ and $\mathbf{C}^x$

[Qing. Comms. Stat Theory and Meth. 46.4 (2017): 1594-1605.]

Gaussian copula generation II

Relation between $\mathbf{C}^z$ and $\mathbf{C}^x$

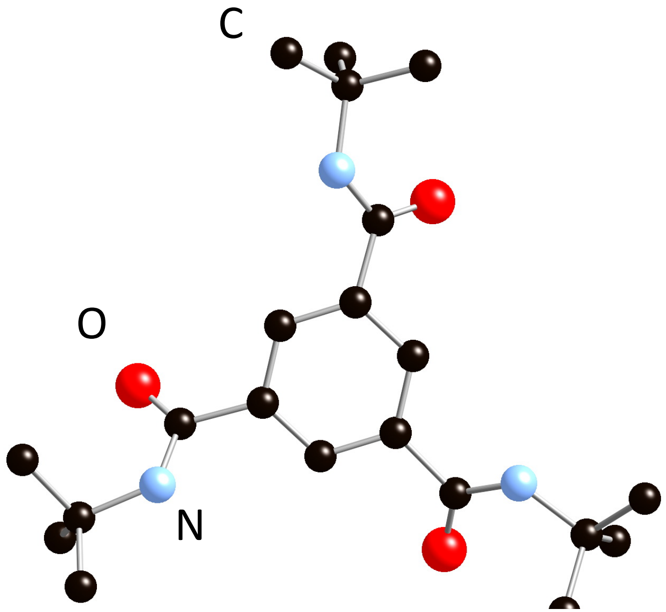
$$C_{ij}^x = \text{Cov}(x_i, x_j) = \int_0^\infty \int_0^\infty \phi_2(z_i, z_j; C_{ij}^z)\, dz_1 dz_2 - P(z_i > 0)P(z_j > 0)$$

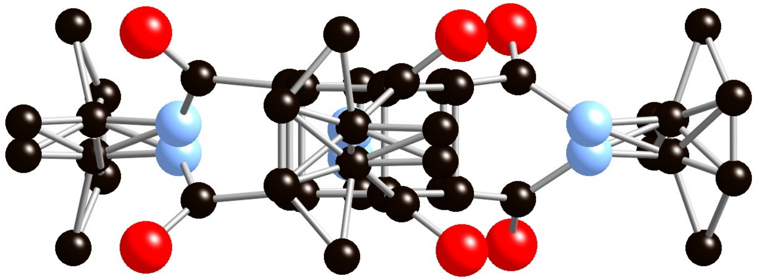
where ϕ_2 is the Gaussian density function.
for the specific case when $x_i \sim \text{Bern}(0.5, C^x)$

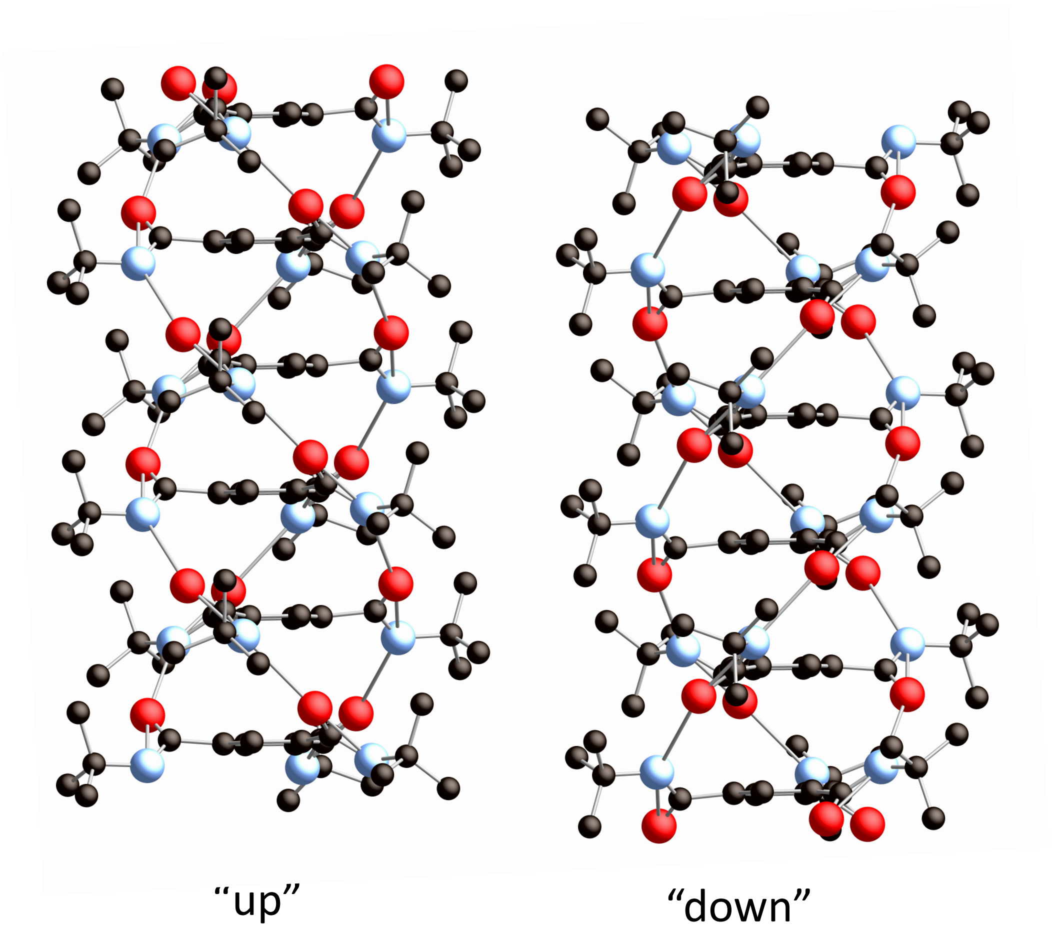
$$C_{ij}^x = \frac{1}{2\pi} \arcsin(C_{ij}^z)$$

[Van Vleck, Middleton. "The spectrum of clipped noise." *Prof. IEEE* 54.1 (2005): 2-19.]

Tris-t-butyl-benzene-tricarboxamide

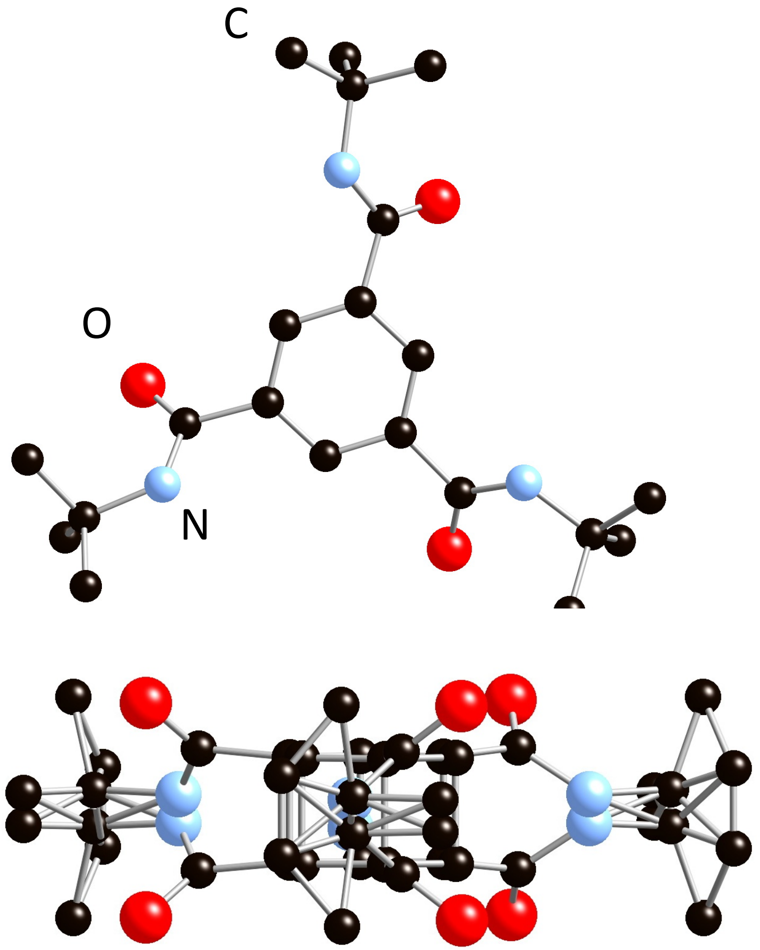




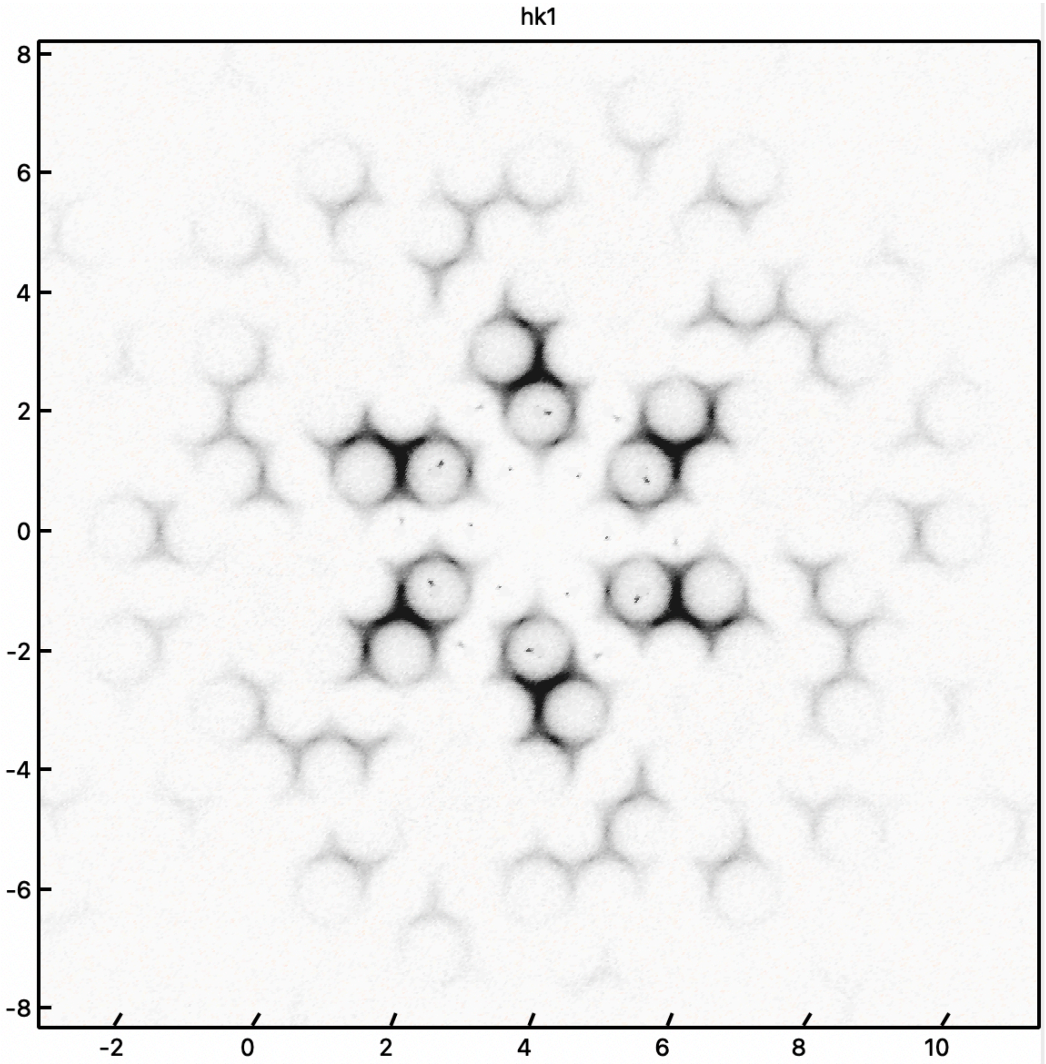


A. Simonov et al. J. Appl. Cryst, 47(6) 2014

Tris-t-butyl-benzene-tricarboxamide

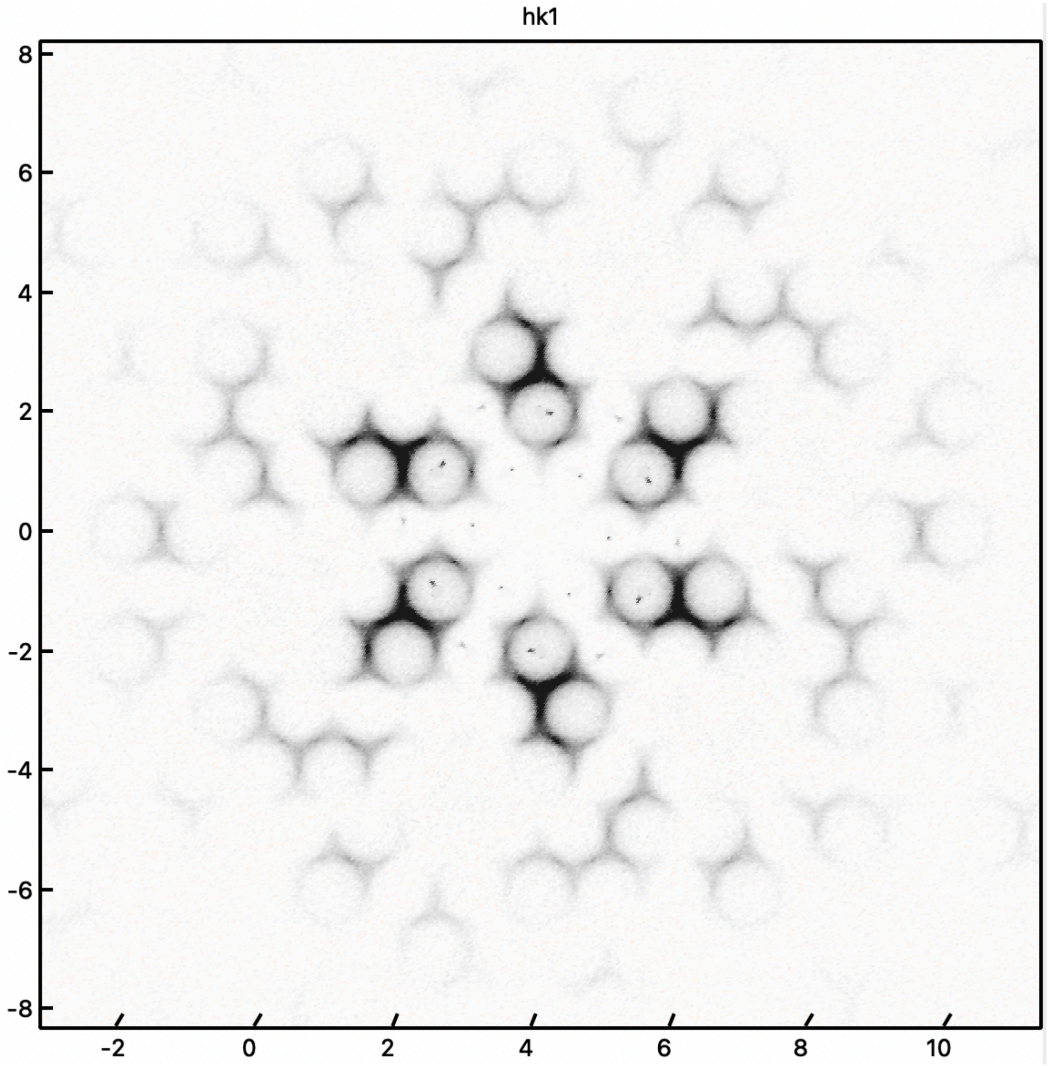


A. Simonov et al. J. Appl. Cryst, 47(6) 2014

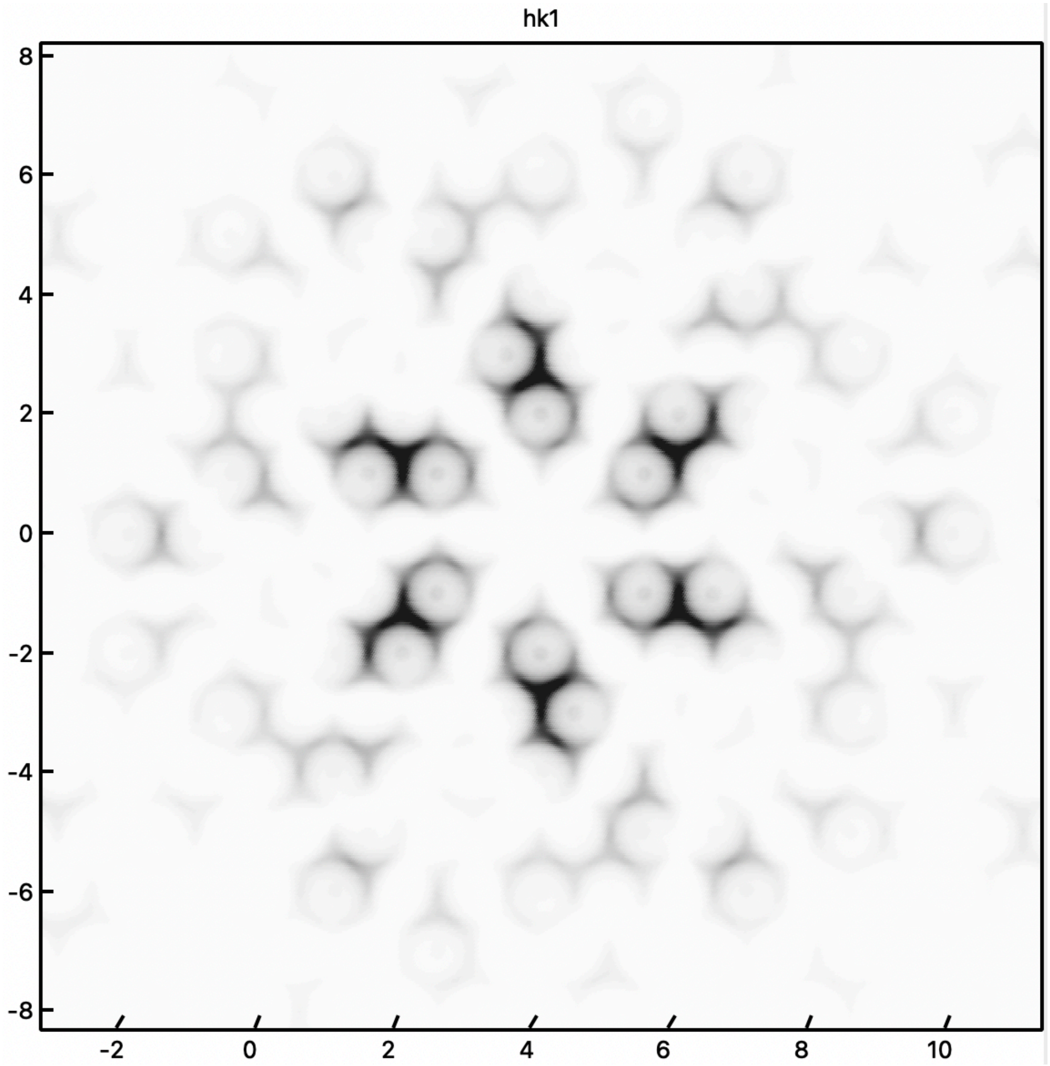


X-ray scattering, hk1 layer

Tris-t-butyl-benzene-tricarboxamide

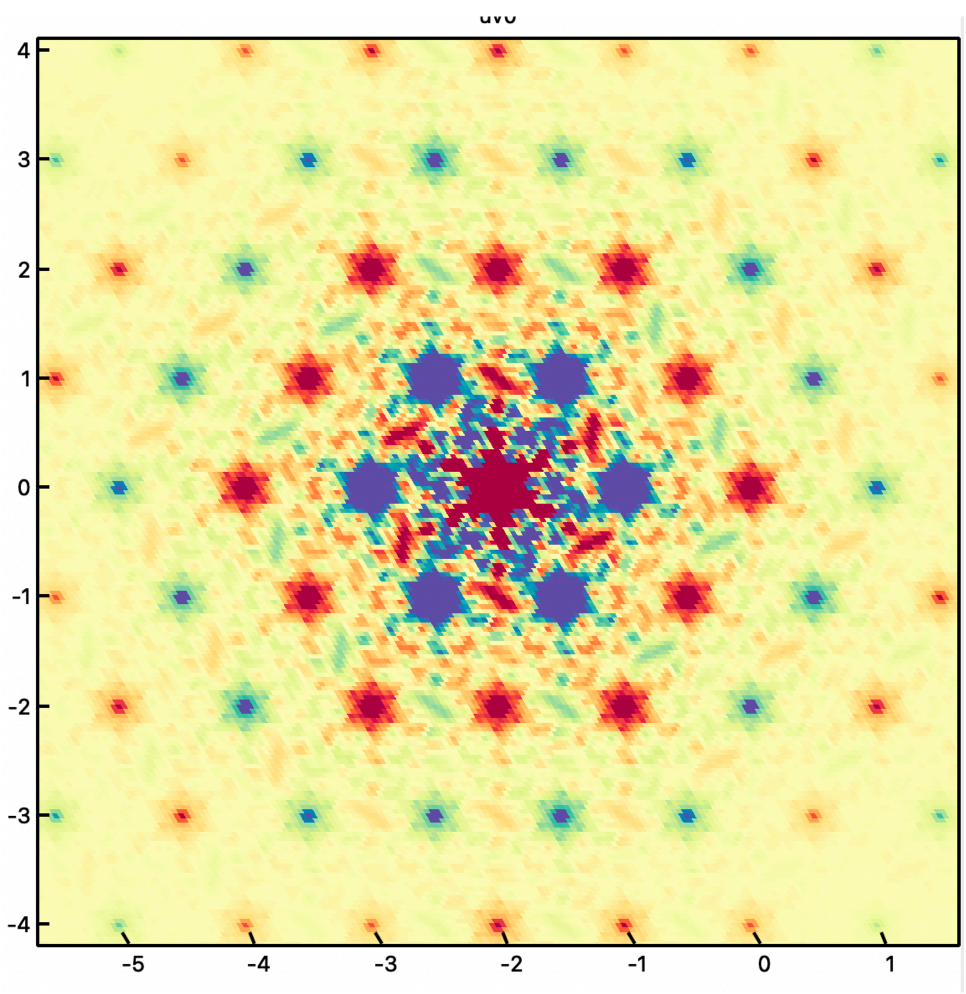


X-ray scattering, hk1 layer

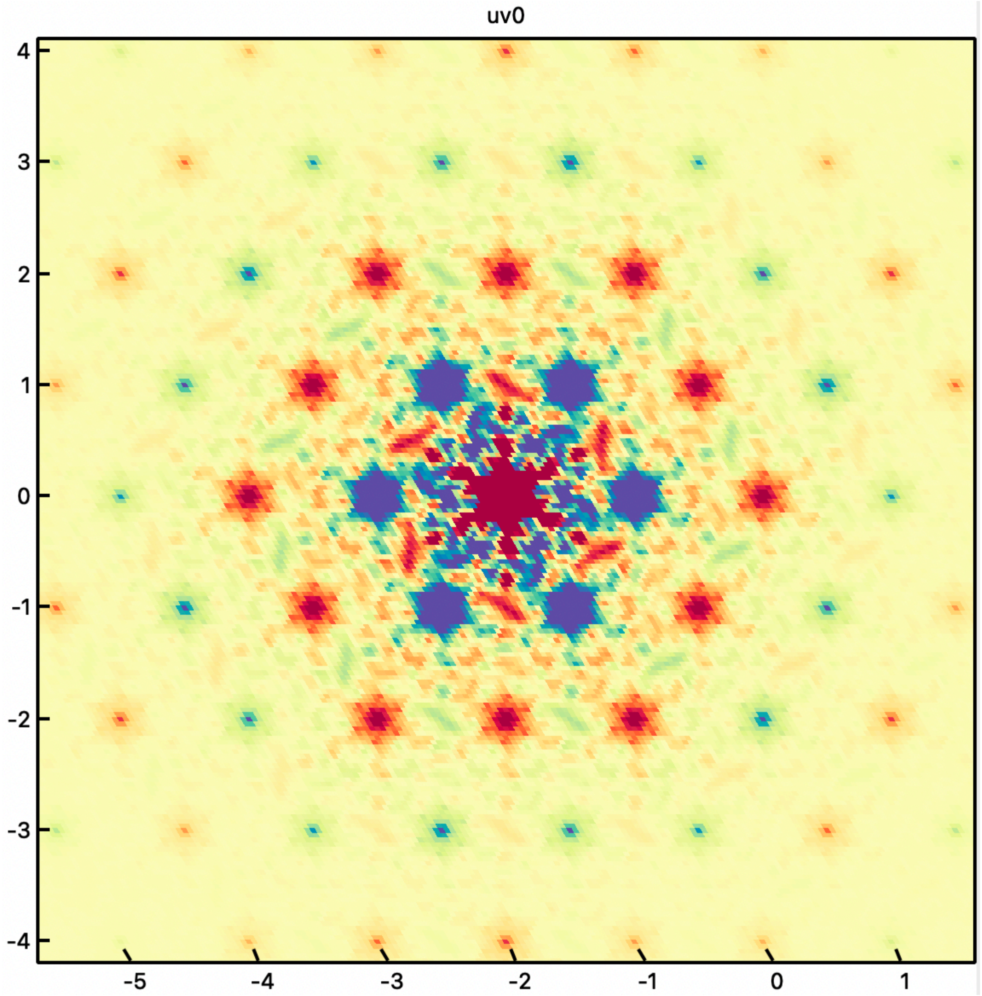


“Soft model”

Experimental ΔPDF



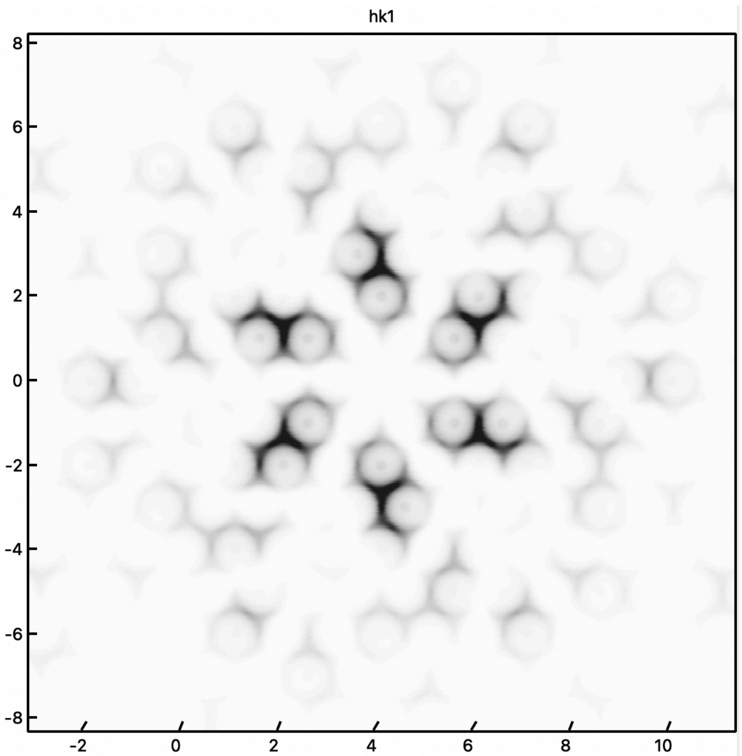
ΔPDF, UV0 cut, experimental

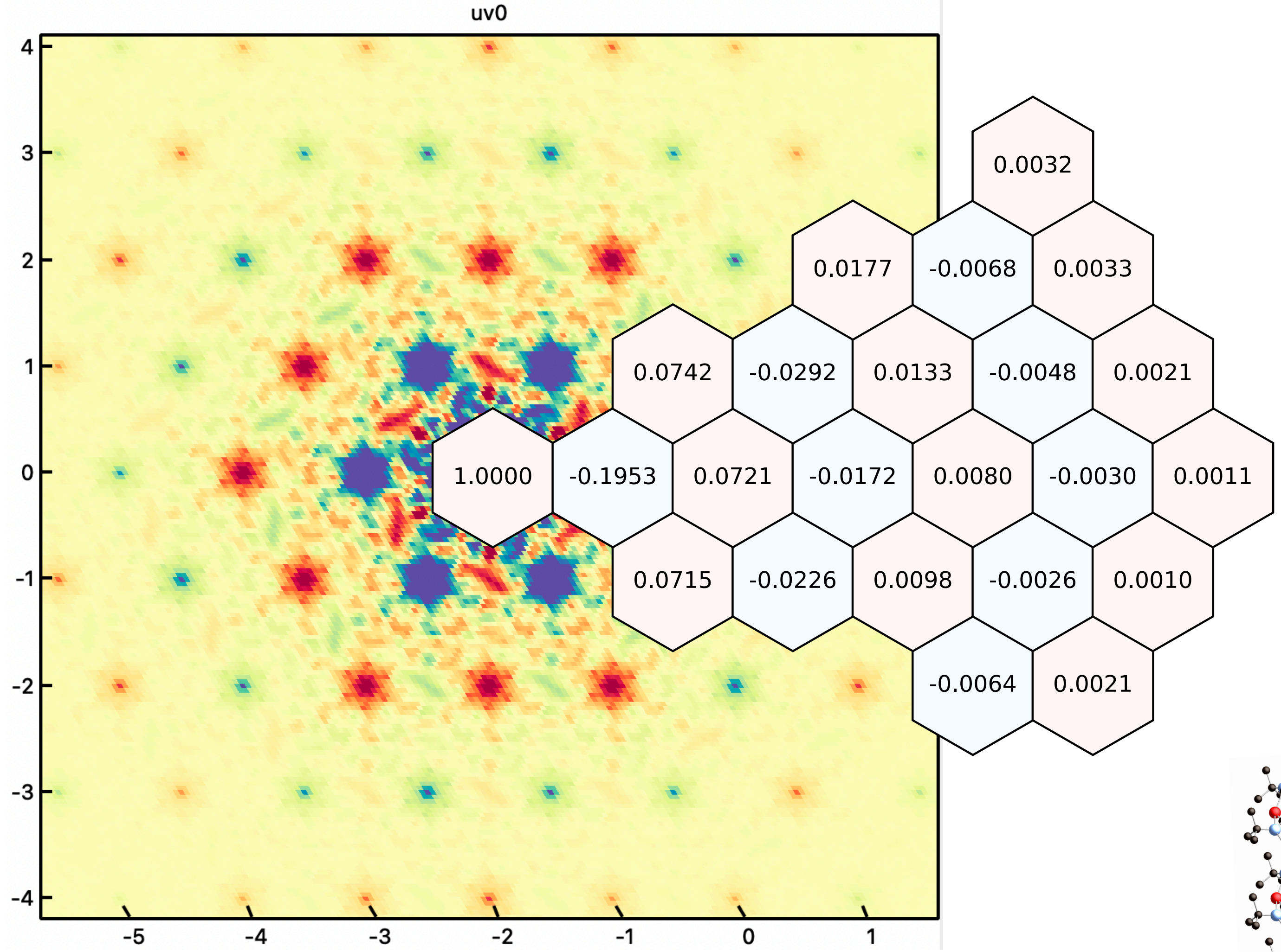


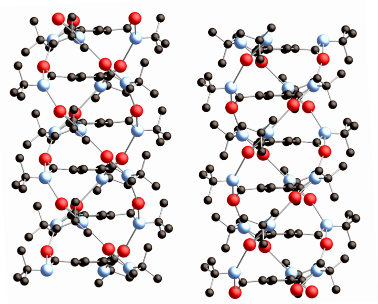
ΔPDF, UV0 cut, soft model

Soft model

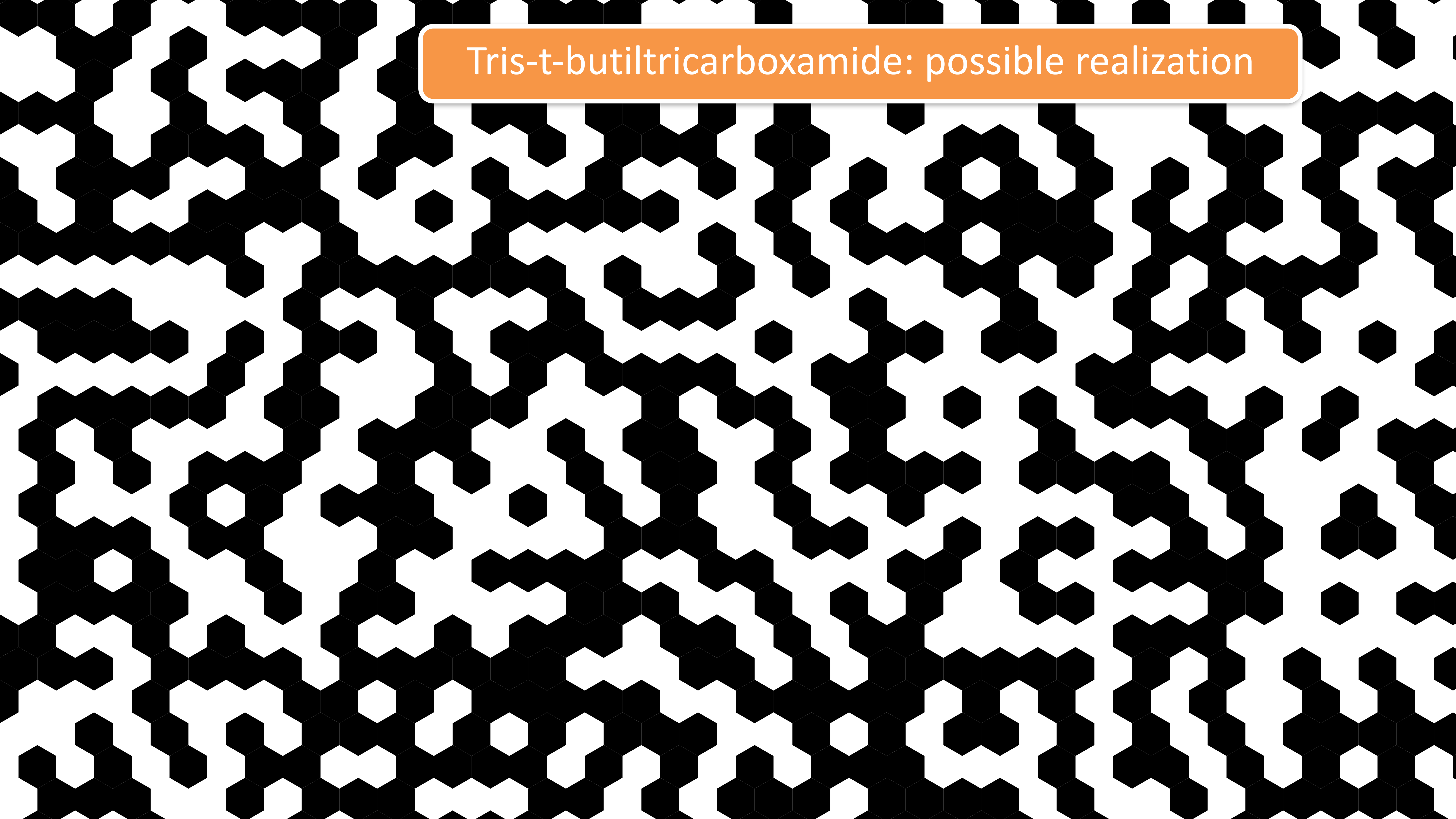
Target correlations, simulation







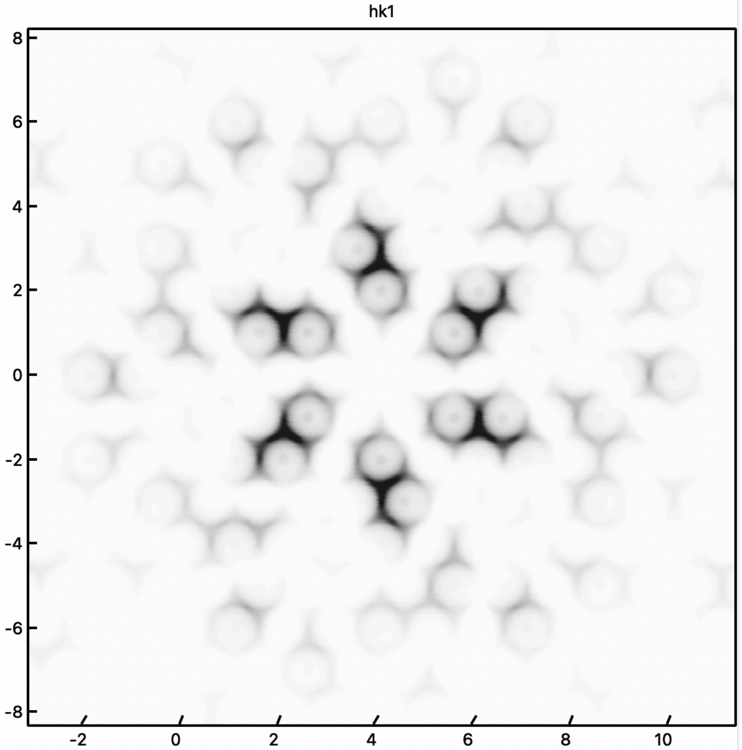
Tris-t-butiltricarboxamide: possible realization

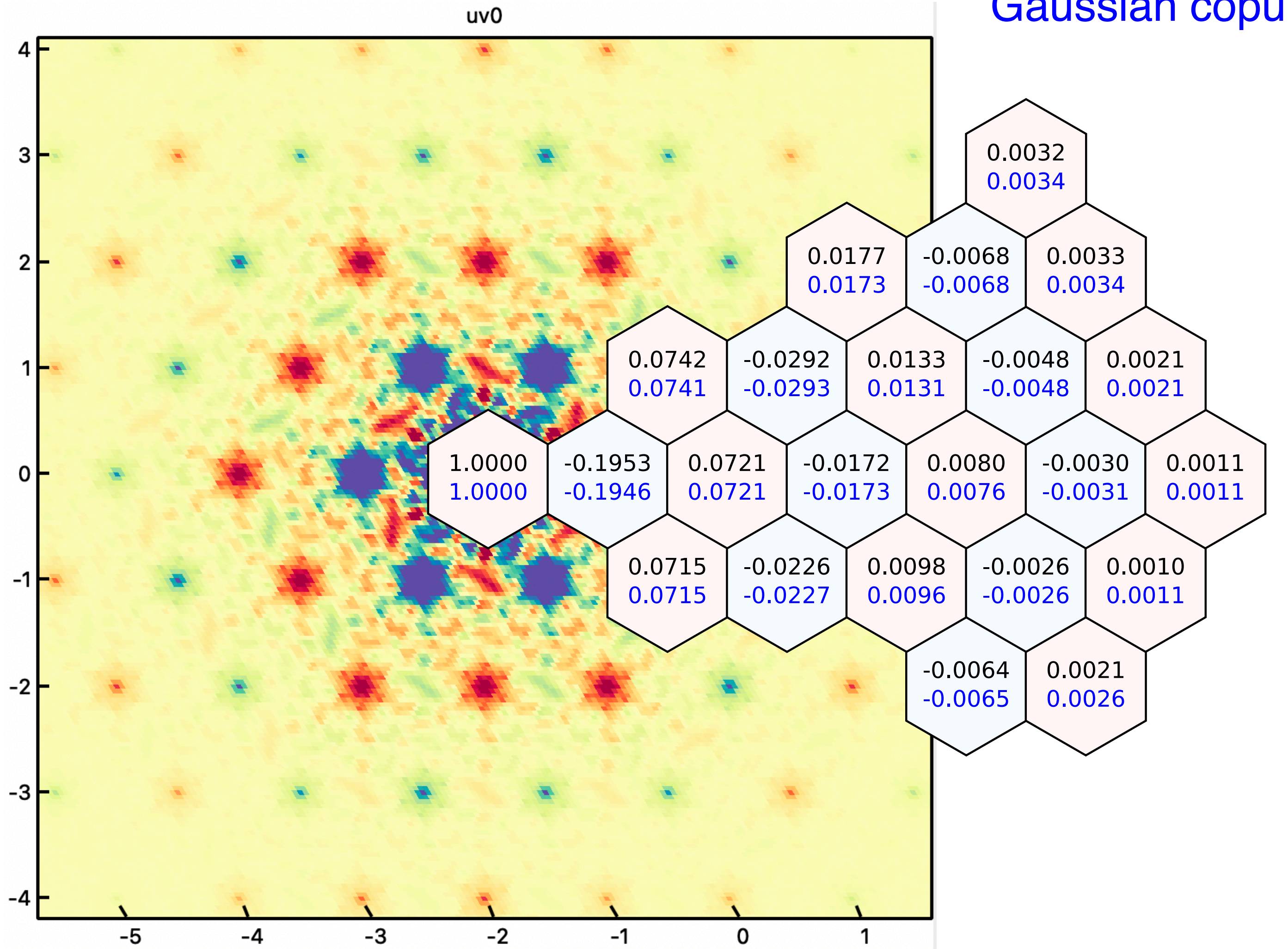


Soft model

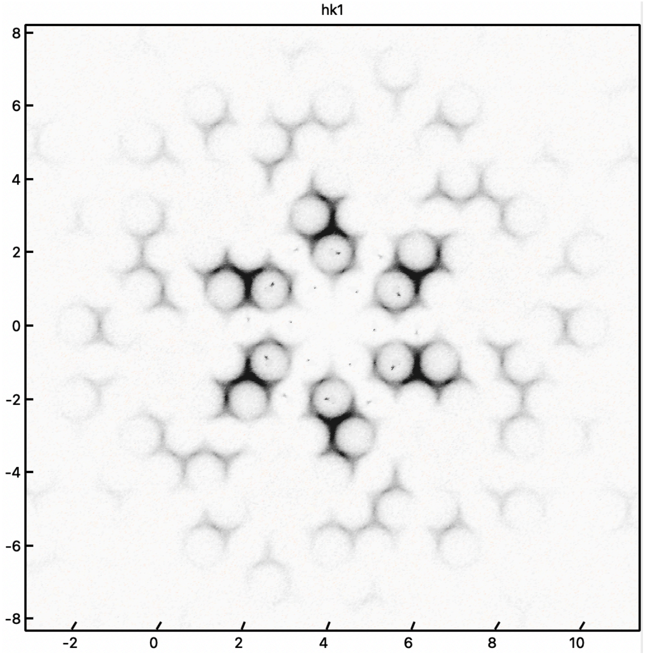
Target correlations, simulation

Gaussian copula correlations



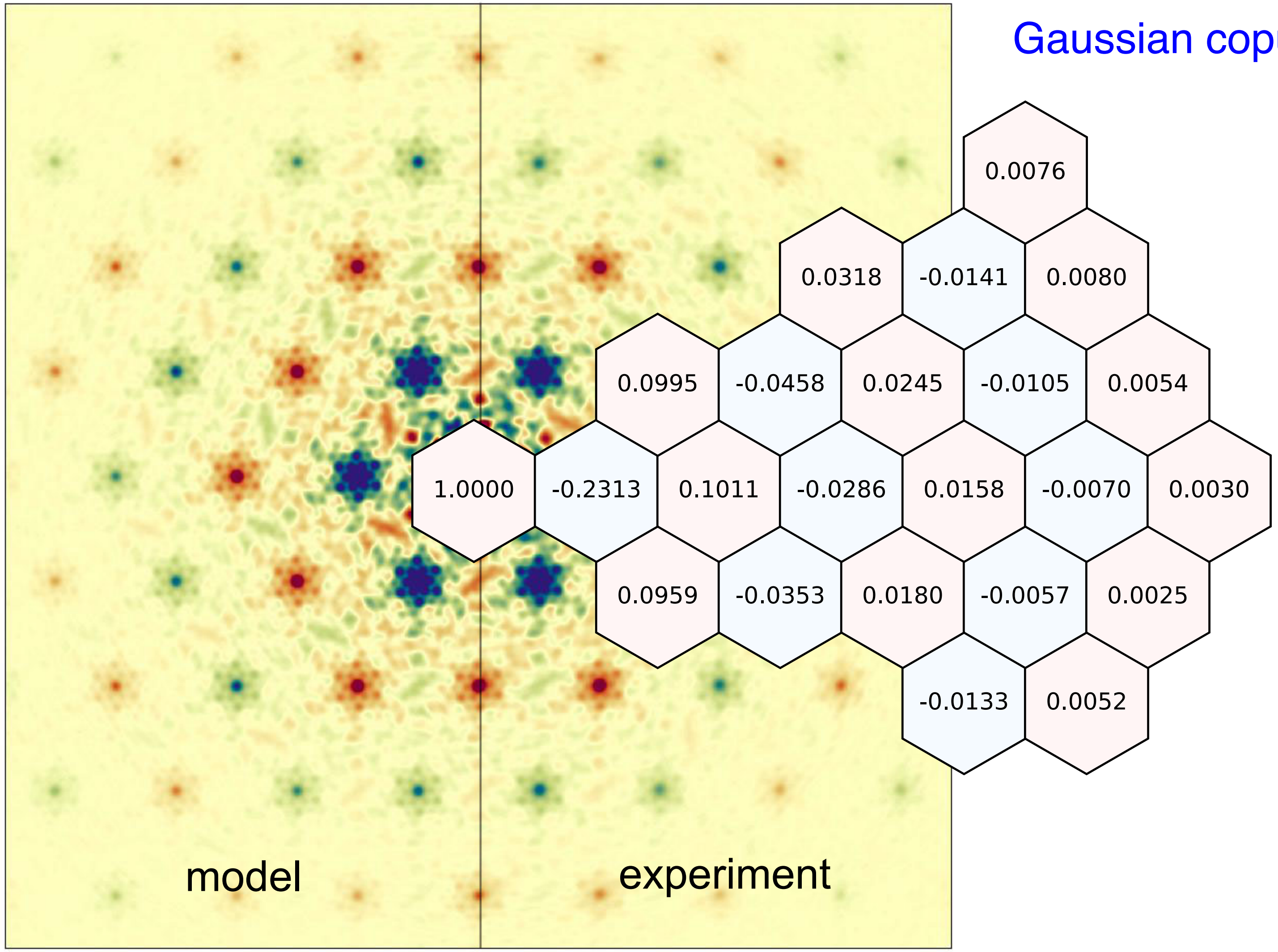


Experimental model*



Target correlations

Gaussian copula correlations

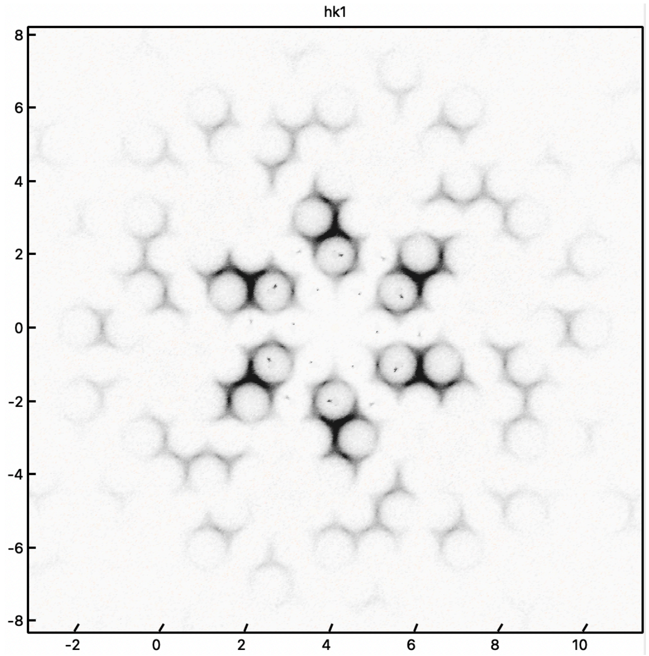


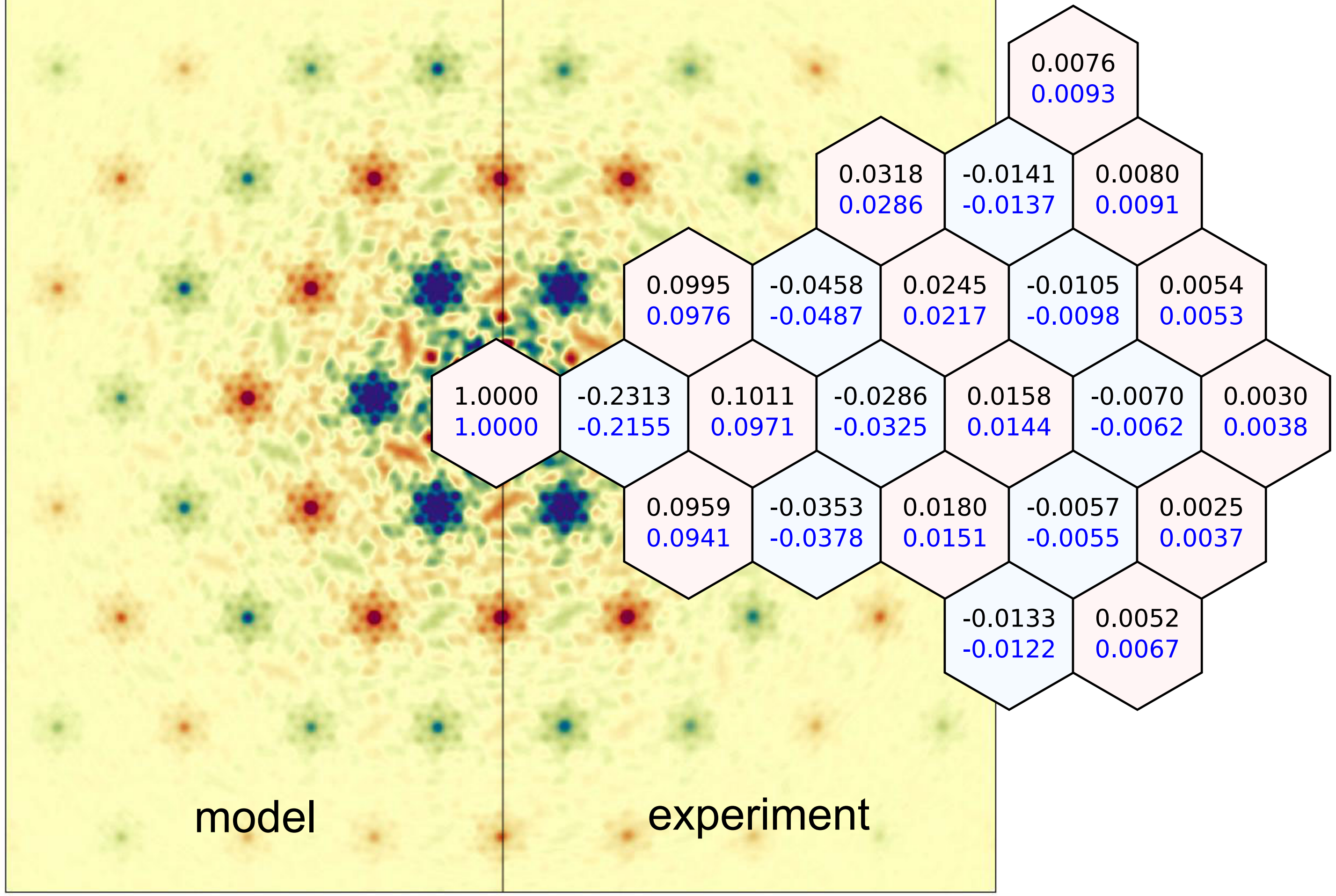
*Values of target correlations obtained by PDF projection

Results

Target correlations

Gaussian copula correlations





Simonov, A., Weber, T., & Steurer, W. (2014).
J. Appl. Cryst., 47(6), 2011-2018.

```python
def generate_target_distribution(Cx):
    Cz = sin(2*pi*Cx/4) #calcualte covariance of Gaussian variables
    Cz_rec = ifftn(Cz)
    Cz_rec[Cz_rec<0]=0 #trim negative eigenvalues to zero
    y = normal(size=shape(conv_kernel)) #generate Gaussian variables y
    #Line below is equivalent to z = Ly but applied efficiently through FFT
    z = real(fftn(sqrt(Cz_rec)*ifftn(y)))
    result = z>0 #threshold
    return result #Final distribution of 0s and 1s
```

```python
def generate_target_distribution(Cx):
    Cz = sin(2*pi*Cx/4) #calcualte covariance of Gaussian variables
    Cz_rec = ifftn(Cz)
    Cz_rec[Cz_rec<0]=0 #trim negative eigenvalues to zero
    y = normal(size=shape(conv_kernel)) #generate Gaussian variables y
    #Line below is equivalent to z = Ly but applied efficiently through FFT
    z = real(fftn(sqrt(Cz_rec)*ifftn(y)))
    result = z>0 #threshold
    return result #Final distribution of 0s and 1s
```

```python
def generate_mc_distribution(grid, steps, J, T):
    for _ in range(steps):
        i = random_site(grid)
        # Calculate energy change dE based on neighbors
        dE = calculate_dE(grid, i, J)

        # Accept or reject flip
        if dE <= 0 or random() < exp(-dE/T):
            grid[i] *= -1
    return grid
```

```python
def generate_target_distribution(Cx):
    Cz = sin(2*pi*Cx/4) #calcualte covariance of Gaussian variables
    Cz_rec = ifftn(Cz)
    Cz_rec[Cz_rec<0]=0 #trim negative eigenvalues to zero
    y = normal(size=shape(conv_kernel)) #generate Gaussian variables y
    #Line below is equivalent to z = Ly but applied efficiently through FFT
    z = real(fftn(sqrt(Cz_rec)*ifftn(y)))
    result = z>0 #threshold
    return result #Final distribution of 0s and 1s
```

```python
%%time
for i in range(10000):
    random_field = generate_target_distribution(Cx)
```

```
CPU times: user 474 ms, sys: 3.97 ms, total: 478 ms
Wall time: 476 ms
```

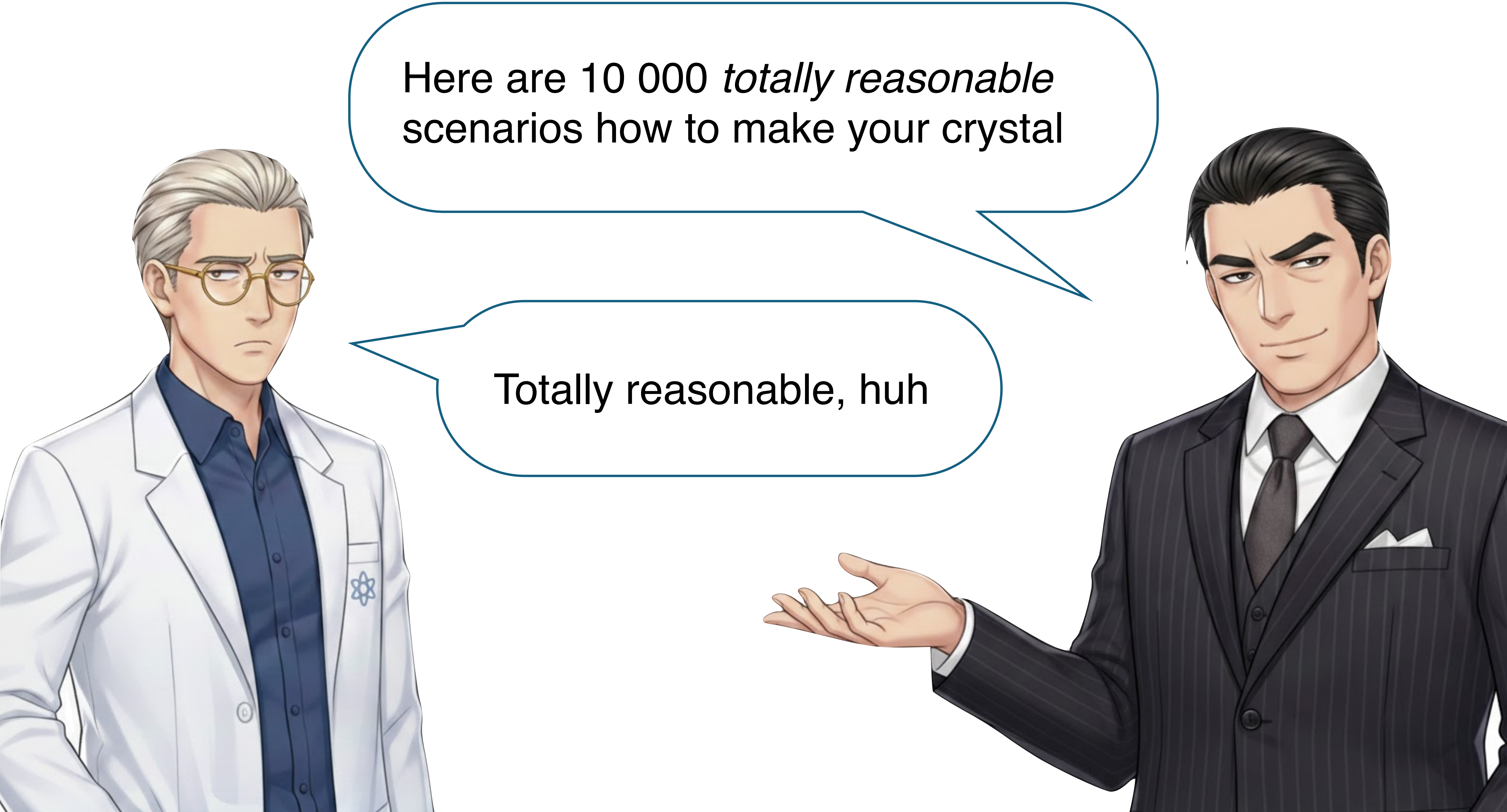
Extensions

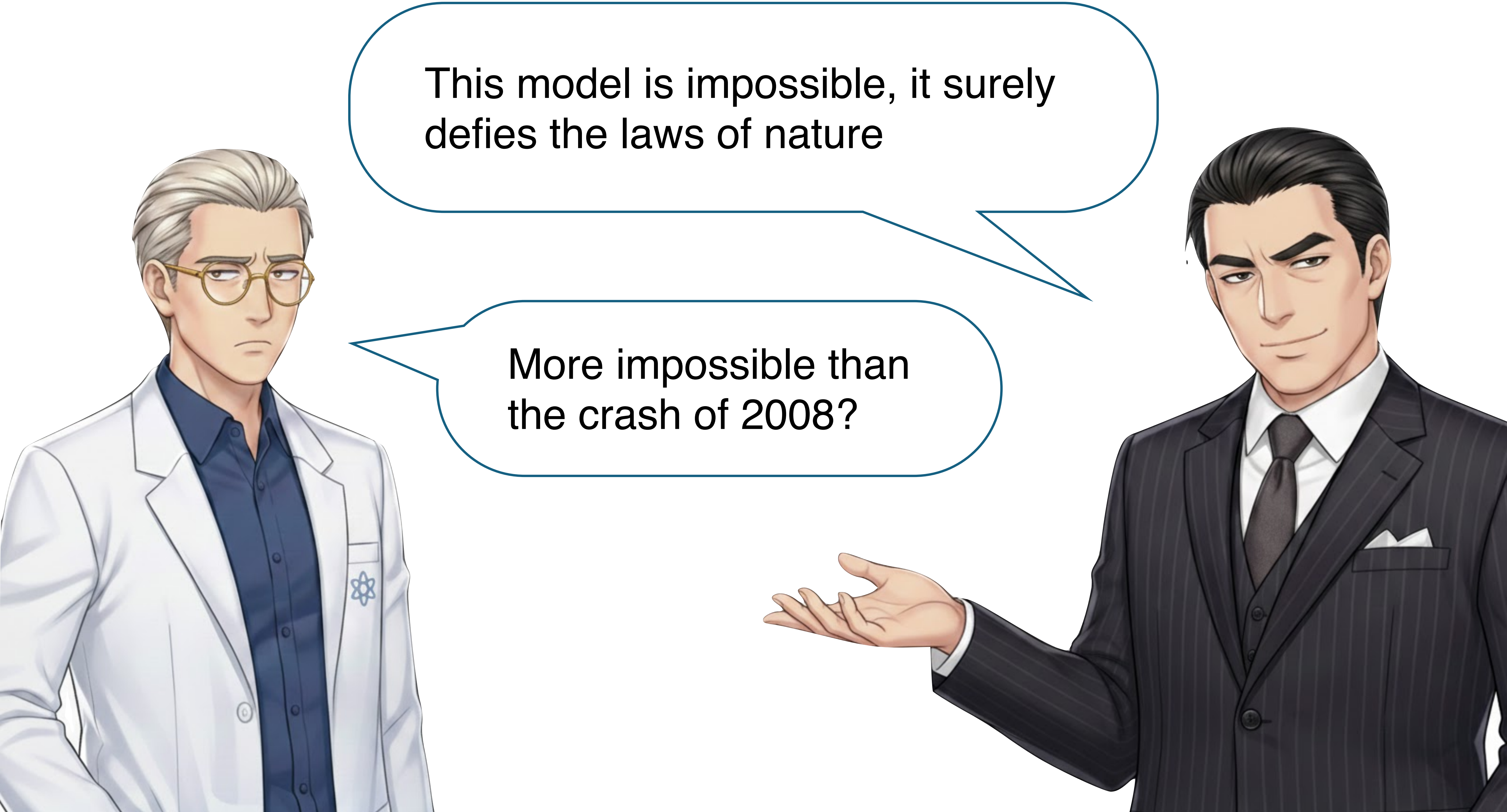
- Arbitrary occupancies (arcsin -> Gaussian integral)
- More than one atom on the site (look for *categorical distribution* in the copula literature)
- Can couple different variables:
  It can model displacements and occupancies at the same time in principle, anyone curious to test how it works?

[Van Vleck, Middleton. "The spectrum of clipped noise." *Prof. IEEE* 54.1 (2005): 2-19.]
[An introduction to Copulas, 2006]
[Caprara, Alberto, et al. *IEEE Trans. Sign. Proc.* 62.6 (2014): 1603-1612.]
[Jacovitti, et. al.*IEEE Trans. Img. Proc.* 7.11 (1998): 1615-1621.]
[Qing. Comms. Stat Theory and Meth. 46.4 (2017): 1594-1605.]
[Bosak, A., et al. *arXiv preprint arXiv:1101.0490* (2011).]

Here is your crystal

Oh this is how it **actually** looks like


Here are 10 000 *totally reasonable* scenarios how to make your crystal
Totally reasonable, huh


This model is impossible, it surely defies the laws of nature
More impossible than the crash of 2008?

"It is hard for us, and without being flippant, to even see a scenario within any kind of realm of reason that would see us losing $1 in any of those transactions,"

Joseph Cassano head of Financial Products Division of AIG, August 2007.

Conclusions

Gaussian Copula is method to generate disordered crystals consistent with 3D-ΔPDF refined correlations: fast, simple, sensitive

Conclusions

Gaussian Copula is method to generate disordered crystals consistent with 3D-ΔPDF refined correlations: fast, simple, sensitive, suspicious