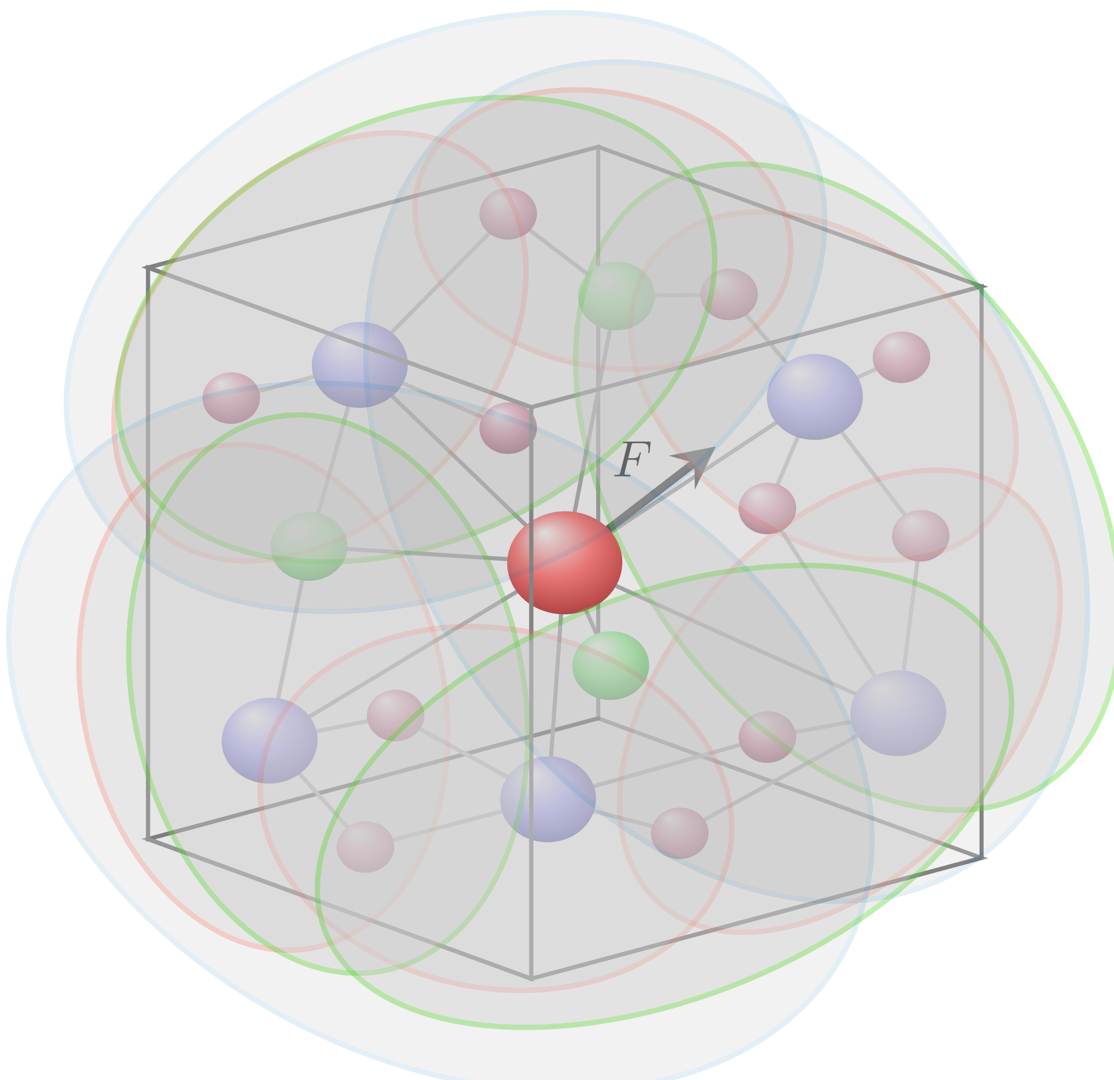
Fourier space neural networks

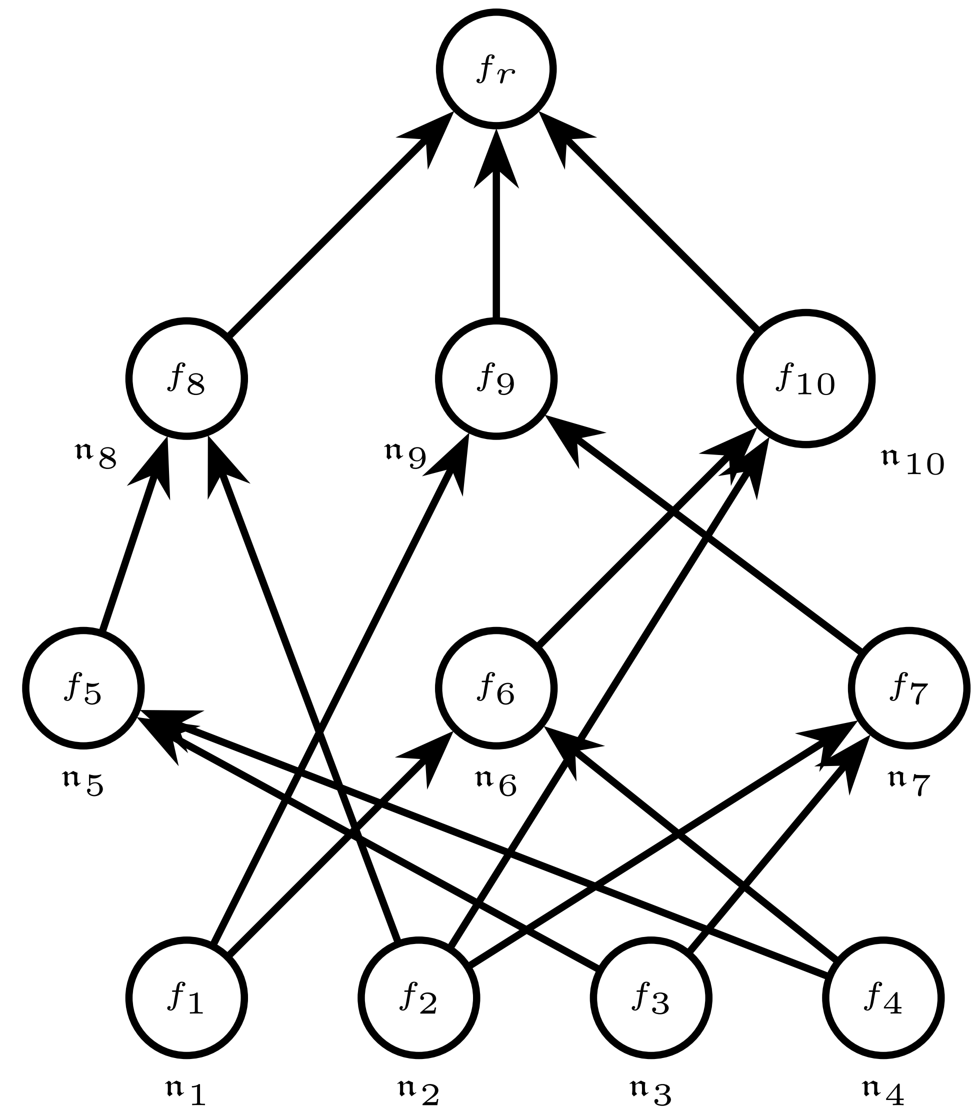
Risi Kondor

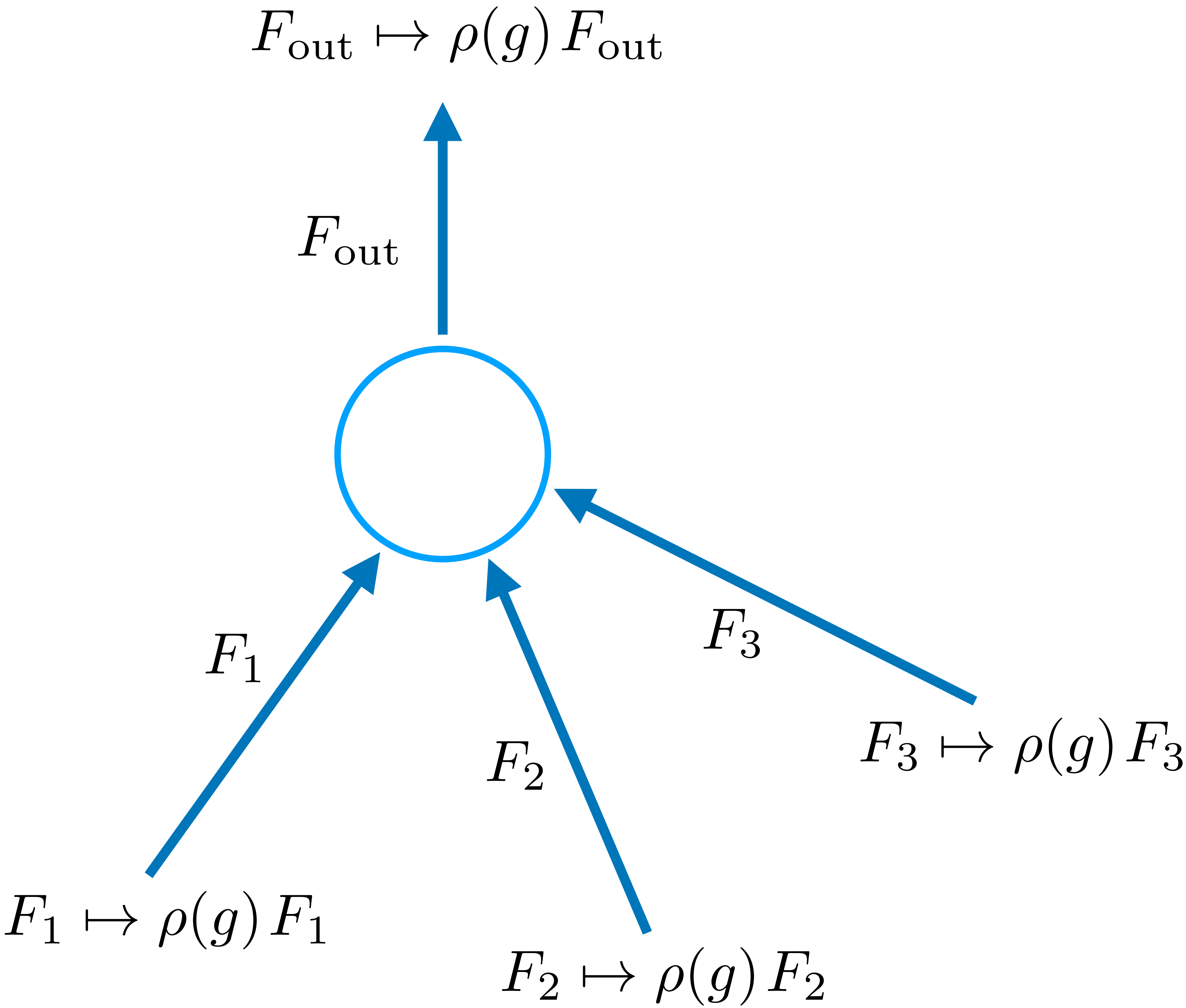
FLATIRON INSTITUTE

The University of Chicago



$$F(\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_m)$$




$F_{\text{out}} \mapsto \rho(g)\, F_{\text{out}}$
F_{out}
F_1
F_2
F_3
$F_1 \mapsto \rho(g)\, F_1$
$F_2 \mapsto \rho(g)\, F_2$
$F_3 \mapsto \rho(g)\, F_3$

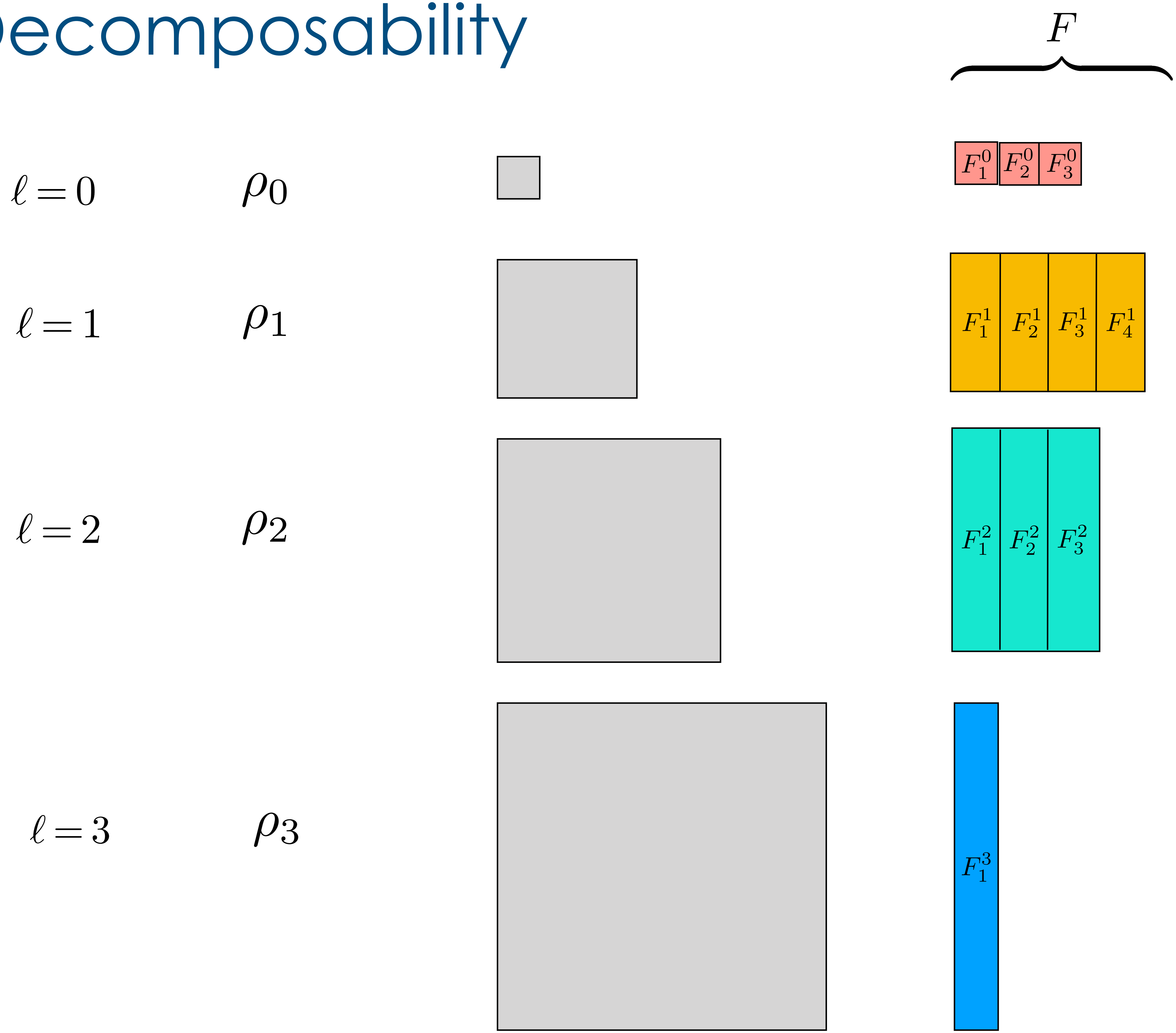
Why representations?

$$F \mapsto \rho(g_1)\, F \mapsto \boxed{\rho(g_2)\,\rho(g_1)\, F = \rho(g_2 g_1)\, F}$$

$$\rho(g) = \quad U \begin{pmatrix} \rho^{(0)}(g) & & & & \\ & \rho^{(0)}(g) & & & \\ & & \rho^{(1)}(g) & & \\ & & & \ddots & \\ & & & & \rho^{(k)}(g) \end{pmatrix} U^\dagger$$

(This works for any finite dimensional representation of any compact group over the complex numbers.)

Decomposability



Group equivariant neural networks (Cohen & Welling, 2016)

Harmonic networks: deep translation and rotation equivariance (Worrall, Garbin, Turmukhanbetov & Brostow, 2016)

Steerable CNNs (Cohen & Welling, 2017)

On the generalization of convolution and equivariance (K & Trivedi, 2018)

Intertwiners between induced representations (Cohen, Geiger & Weiler, 2018)

3D steerable neural networks (Weiler, Geiger, Wellig, Boomsma & Cohen, 2018)

Gauge equivariant neural networks (Cohen, Weiler, Kicanaoglu, Welling, 2019)

Tensor field networks (Thomas, Smidt, Kearns, Yang, Li Kohlhoff & Riley, 2018)

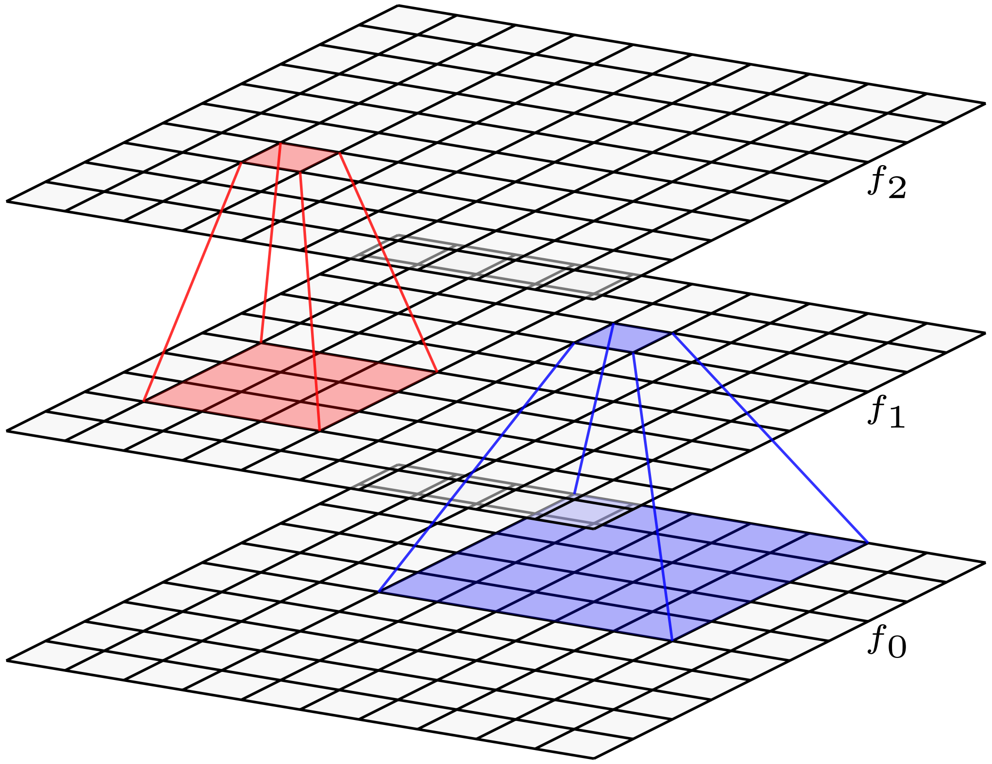
N-body networks (K, 2018)

Cormorant (Anderson, Hy & K, 2019)

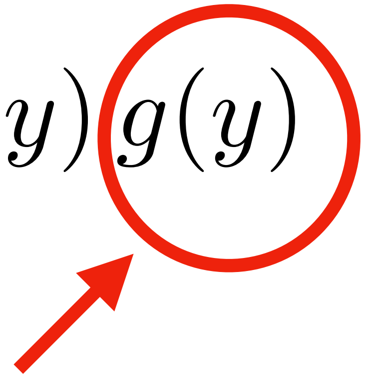
1. Convolution
2. Linear part
3. Nonlinear part
4. Dirty Details
5. Cormorant

1. Convolution

Convolution

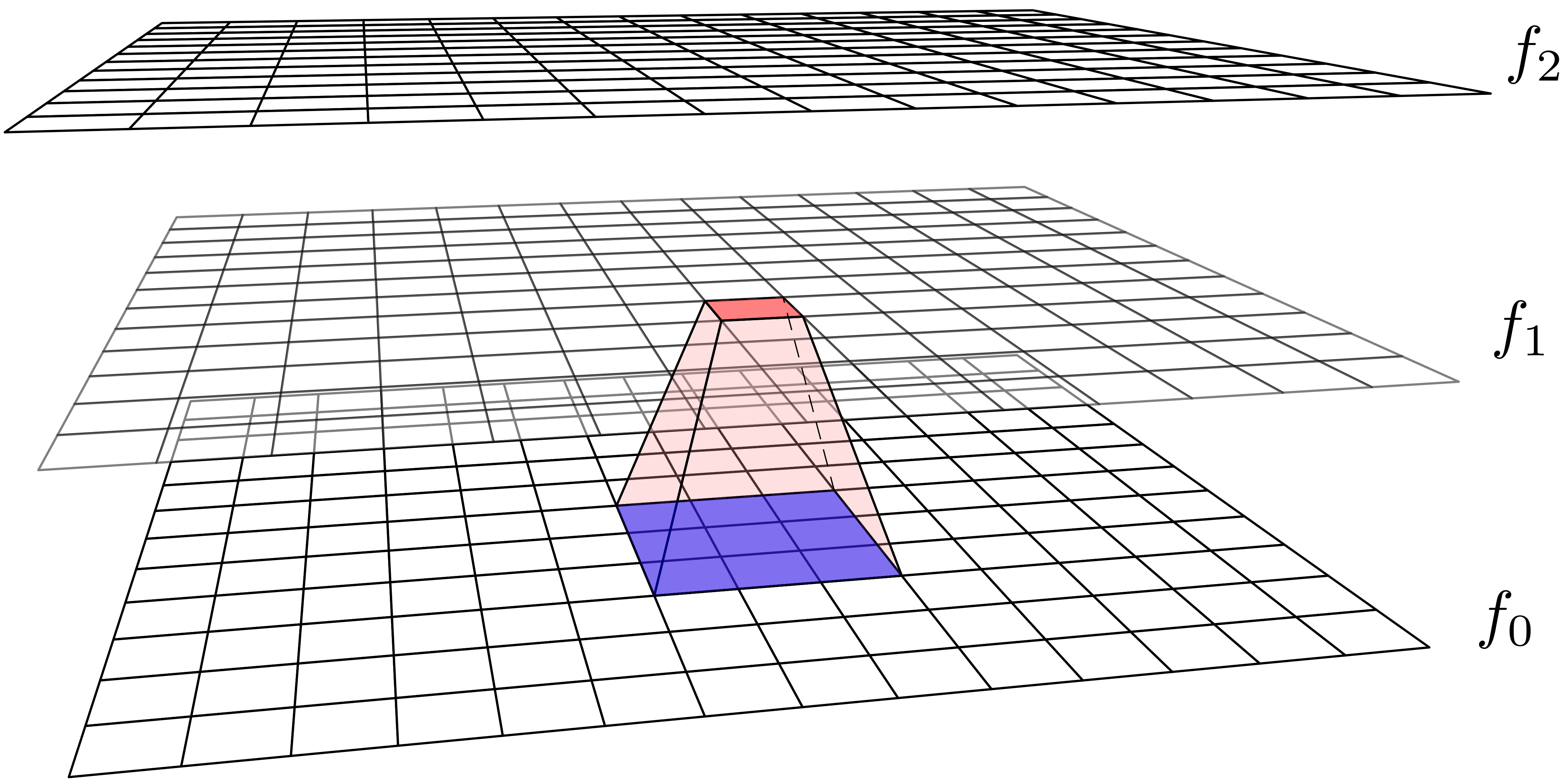


$$(f * g)(x) = \sum_{y \in \mathbb{Z}^2} f(x - y)\, g(y)$$

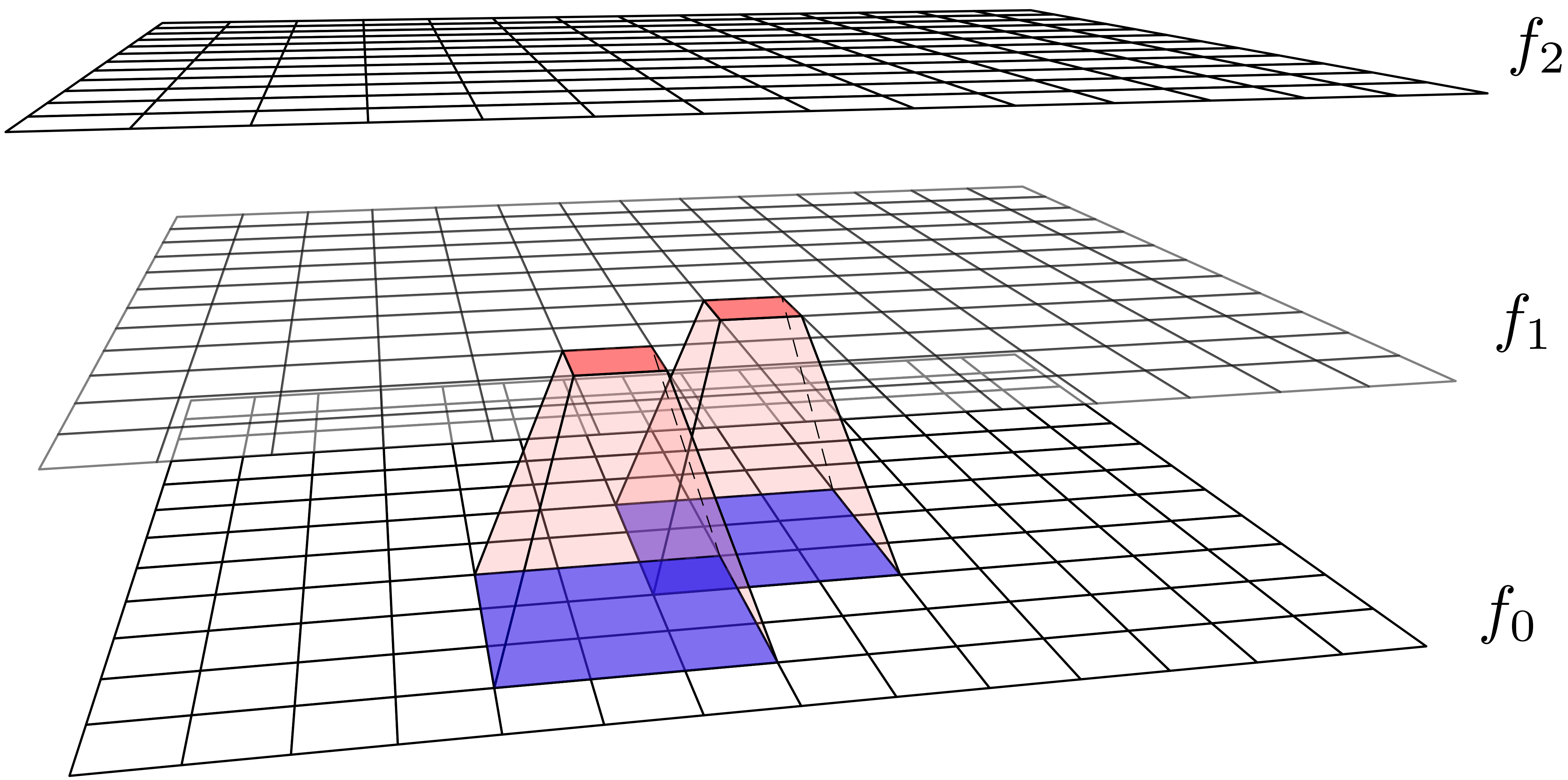


Filter at layer ℓ

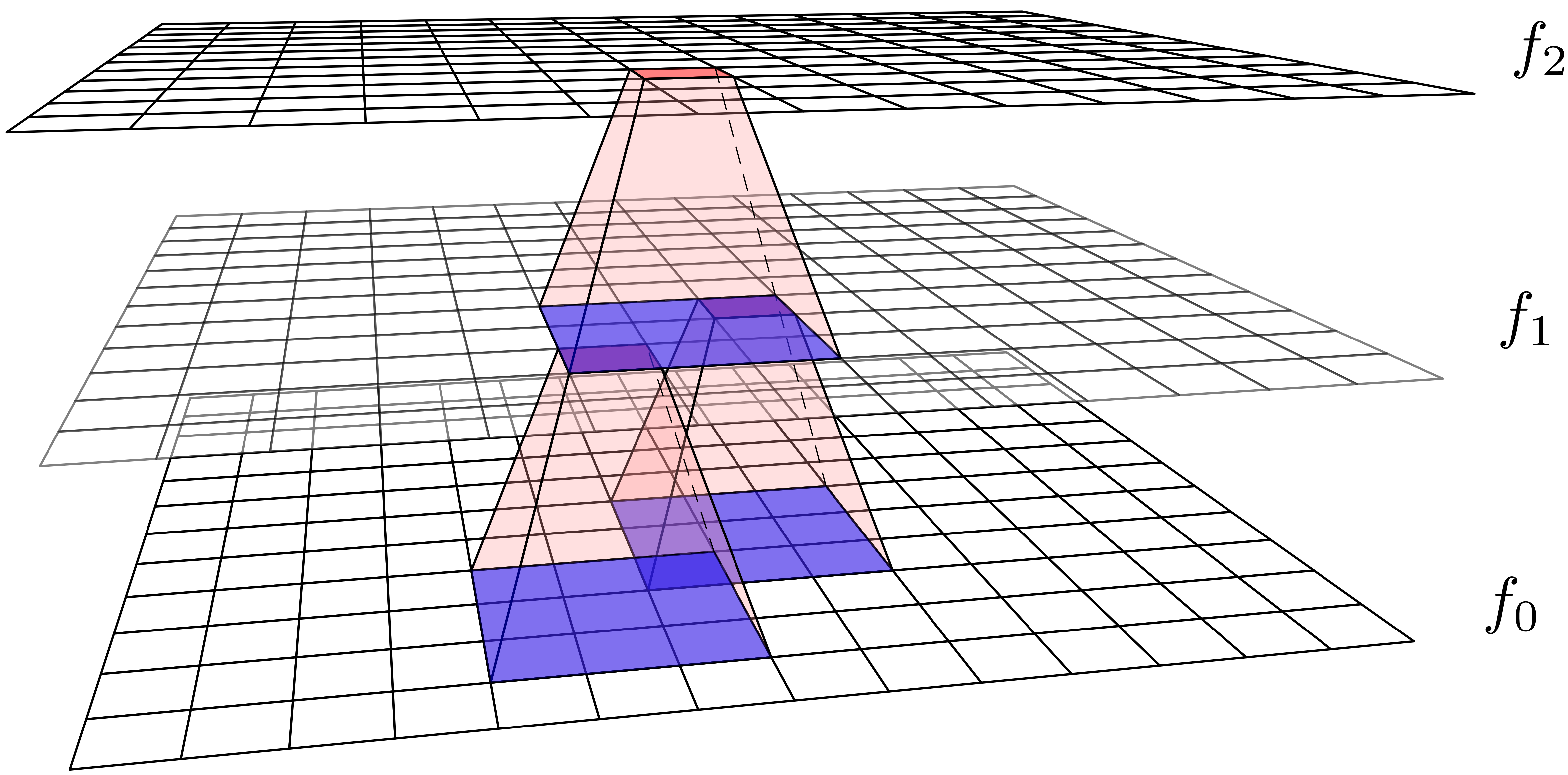
Multiscale structure



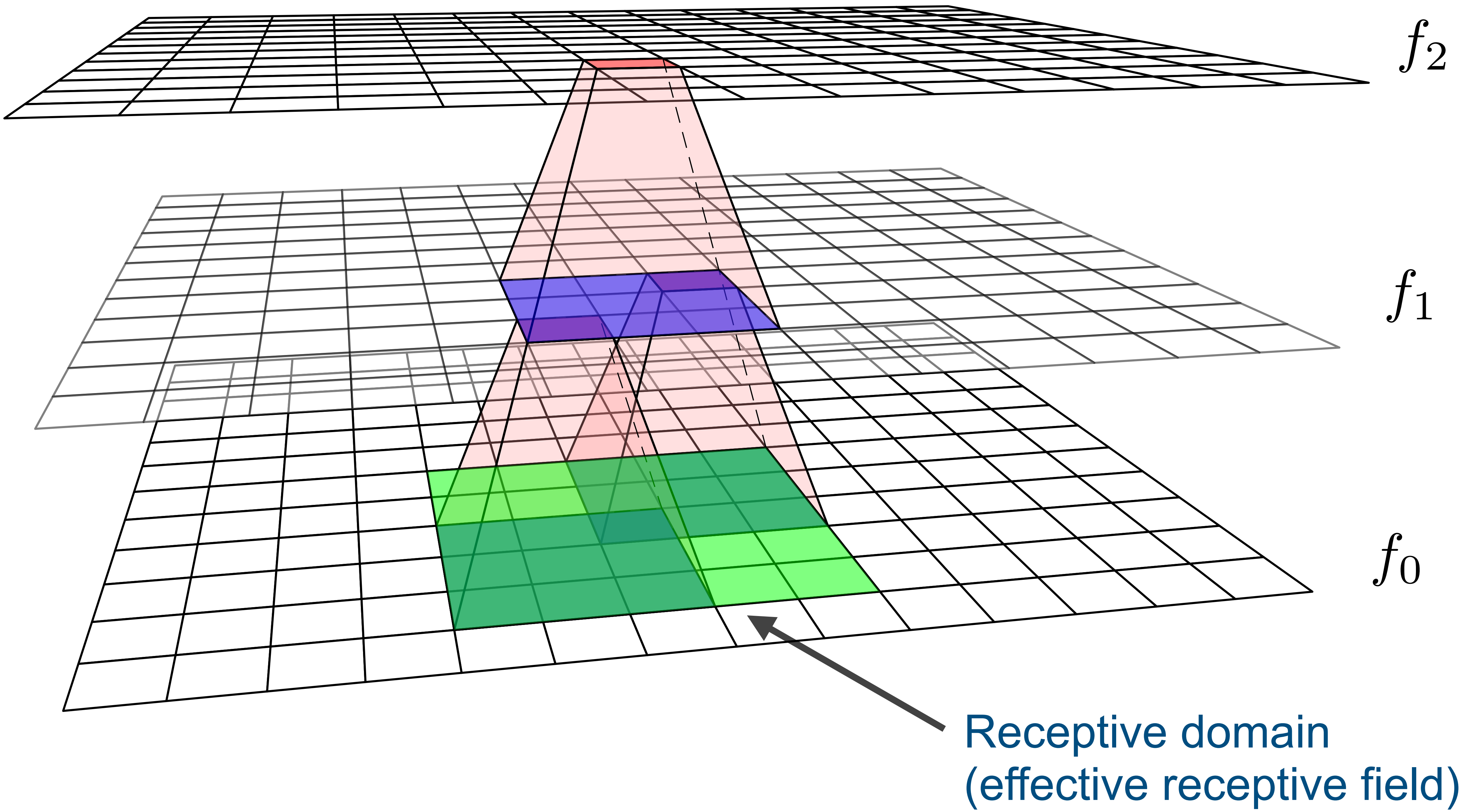
Multiscale structure



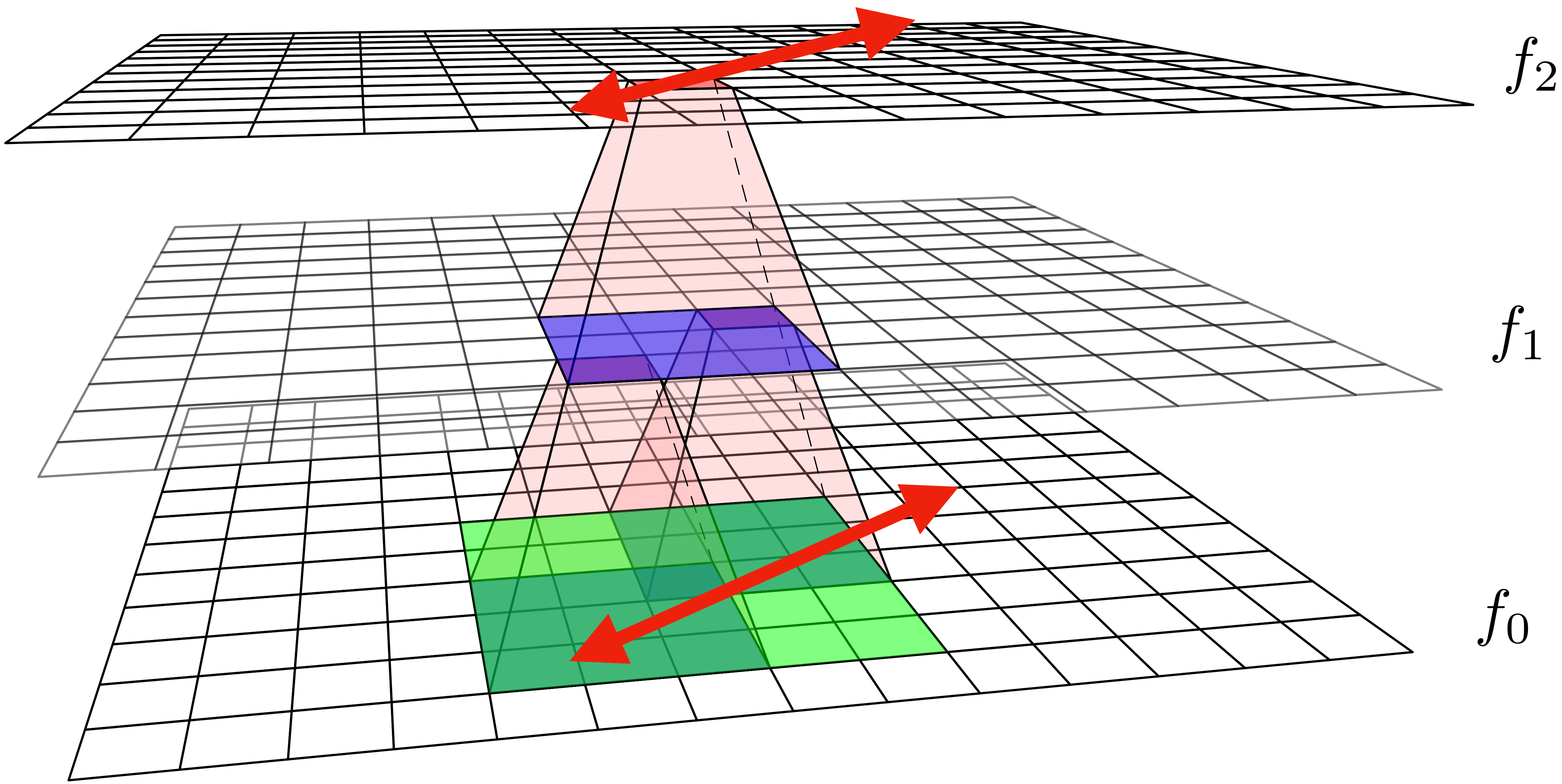
Multiscale structure



Multiscale structure

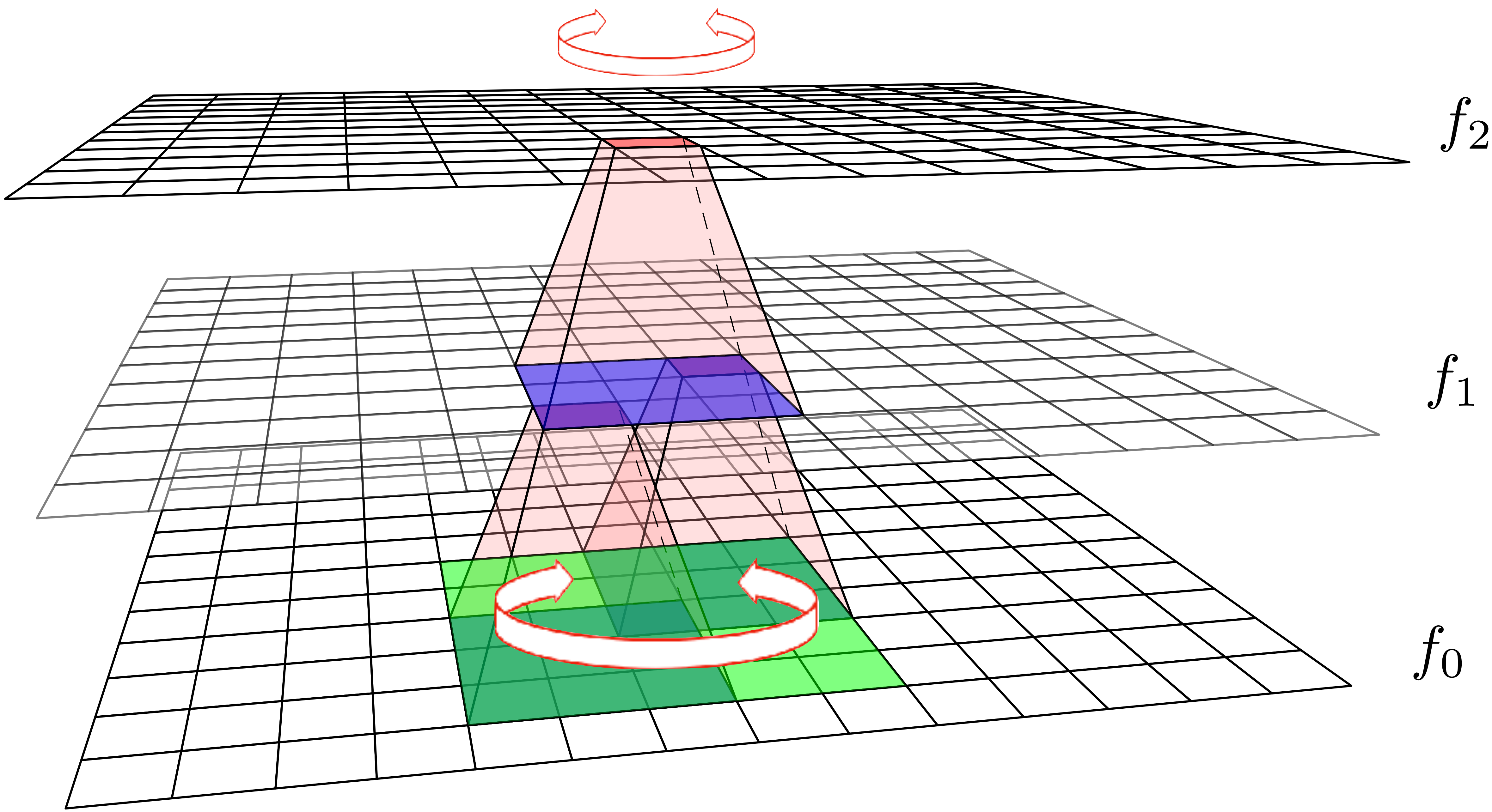


Equivariance (covariance)



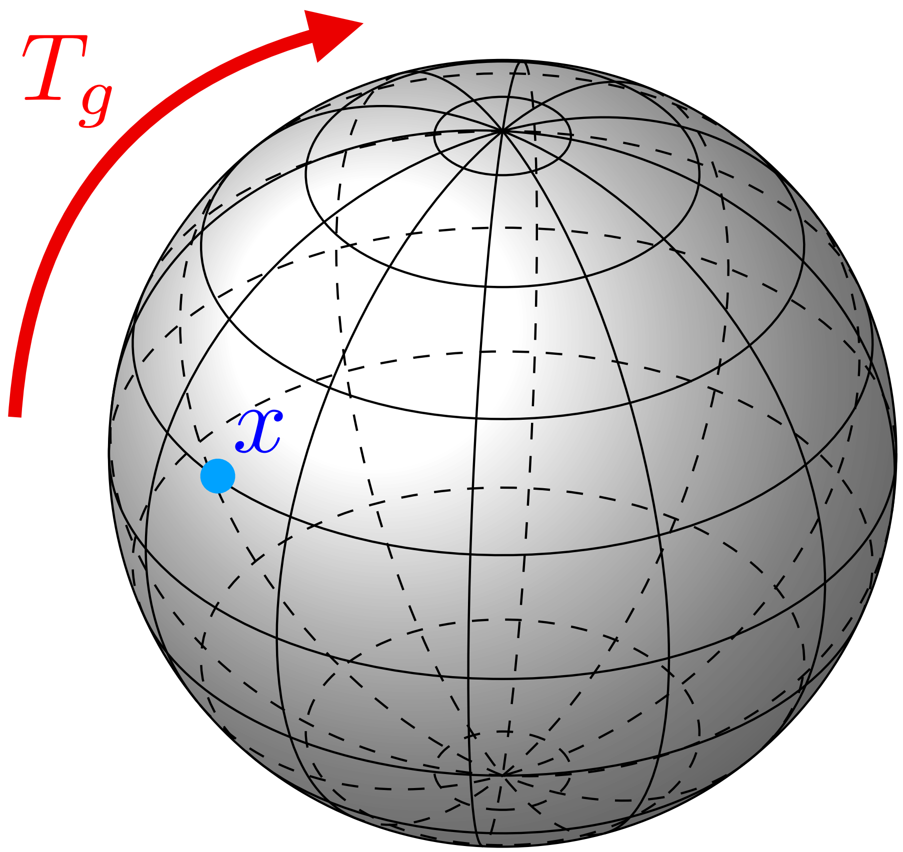
[Cohen & Welling: Group equivariant CNNs (ICML 2016)]

Equivariance (covariance)



[Cohen & Welling: Steerable CNNs (ICLR 2017)]

2. The linear part (on groups)



1. Our function lives on a space $\mathcal{X}$

$$f : \mathcal{X} \to \mathbb{C}$$

2. We have a group G acting on $\mathcal{X}$

$$x \mapsto T_g(x)$$

3. This induces an action on functions

$$f \xmapsto{\mathbb{T}_g} f' \qquad f'(x) = f(T_g^{-1}(x))$$

Translation:

$$f'(x) = f(x - t) \qquad t \in \mathbb{R}^2$$

$$f'(u) = f(t^{-1}u) \qquad t \in G$$

Convolution:

$$(f * g)(x) = \int f(x - y)\, g(y)\, dy$$

$$(f * g)(u) = \int_G f(uv^{-1})\, g(v)\, d\mu(v)$$

Equivariance:

$$(f' * g)(x) = \int f(x - t - y)\, g(y)\, dy =$$
$$= (f * g)(x - t)$$

Translation:

$$f'(x) = f(x-t) \qquad t \in \mathbb{R}^2$$

$$f'(u) = f(t^{-1}u) \qquad t \in G$$

Convolution:

$$(f * g)(x) = \int f(x-y)\, g(y)\, dy$$

$$(f * g)(u) = \int_G f(uv^{-1})\, g(v)\, d\mu(v)$$

Equivariance:

$$(f' * g)(x) = \int f(x-t-y)\, g(y)\, dy = \\ = (f * g)(x-t)$$

$$(f' * g)(u) = \int_G f(t^{-1}(uv^{-1}))\, g(v)\, dv = \\ \int_G f((t^{-1}u)v^{-1})\, g(v)\, dv = (f * g)(t^{-1}u)$$

Fourier transform:

$$\widehat{f}(k) = \int f(x)\, e^{-ikx}\, dx$$

$$\widehat{f}(\rho) = \int f(x)\, \rho(x)\, d\mu(x)$$

A representation of G is a function

$$\rho\colon G \to \mathbb{C}^{d\times d}$$

such that

$$\rho(uv) = \rho(u)\,\rho(v)$$

Fourier transform:

$$\widehat{f}(k) = \int f(x)\, e^{-ikx}\, dx$$

$$\widehat{f}(\rho) = \int f(x)\, \rho(x)\, d\mu(x)$$

Convolution theorem:

$$\widehat{f * g}(k) = \widehat{f}(k) \cdot \widehat{g}(k)$$

$$\widehat{f * g}(\rho) = \widehat{f}(\rho)\, \widehat{g}(\rho)$$

Equivariance:

$$\widehat{f'}(k) = e^{-ikt}\, \widehat{f}(k)$$

$$\widehat{f' * g}(k) = e^{-ikt}\, \widehat{f * g}(k)$$

$$\widehat{f'}(\rho) = \rho(t)\, \widehat{f}(\rho)$$

$$\widehat{f' * g}(\rho) = \rho(t)\, \widehat{f * g}(\rho)$$

Theorem

A feed-forward neural network is equivariant to the action of a compact group G if and only if the linear operation in each layer is of the form

$$\phi_\ell(f_{\ell-1}) = f_{\ell-1} * g_\ell.$$

where $*$ denotes the generalization of convolution to homogeneous space of compact groups, defined

$$(f * g)(u) = \int_G f{\uparrow}^G(uv^{-1})\, g{\uparrow}^G(v)\, d\mu(v)$$

[K and Trivedi, 2018]

$\rho_0 \qquad \rho_1 \qquad \rho_2 \qquad \rho_p$

$$L(\mathcal{X}) = V_0 \oplus V_1 \oplus V_2 \oplus \ldots \oplus V_p$$

$$V_i = W_i^1 \oplus W_i^2 \oplus \ldots \oplus W_i^{m_i}$$

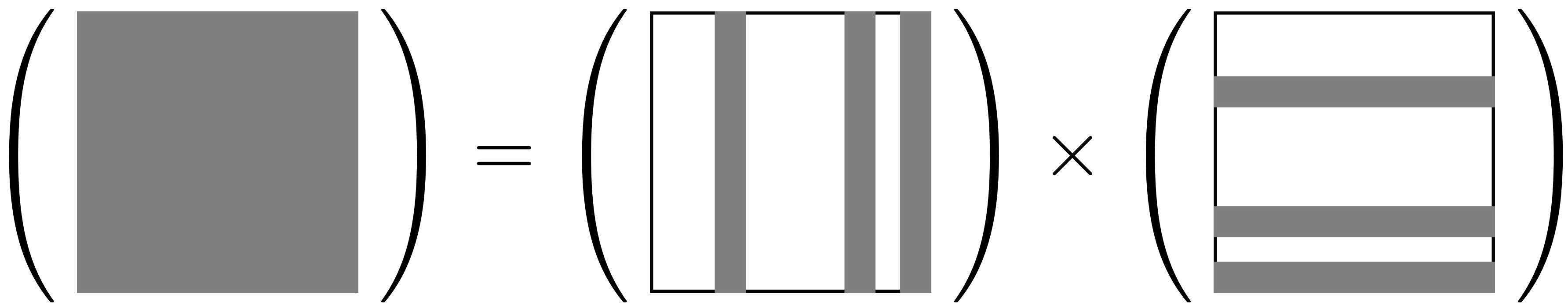
$$\widehat{f * g}(\rho_i) = \widehat{f}(\rho_i) \cdot \widehat{g}(\rho_i)$$

↑

matrix multiplication

Case 1:

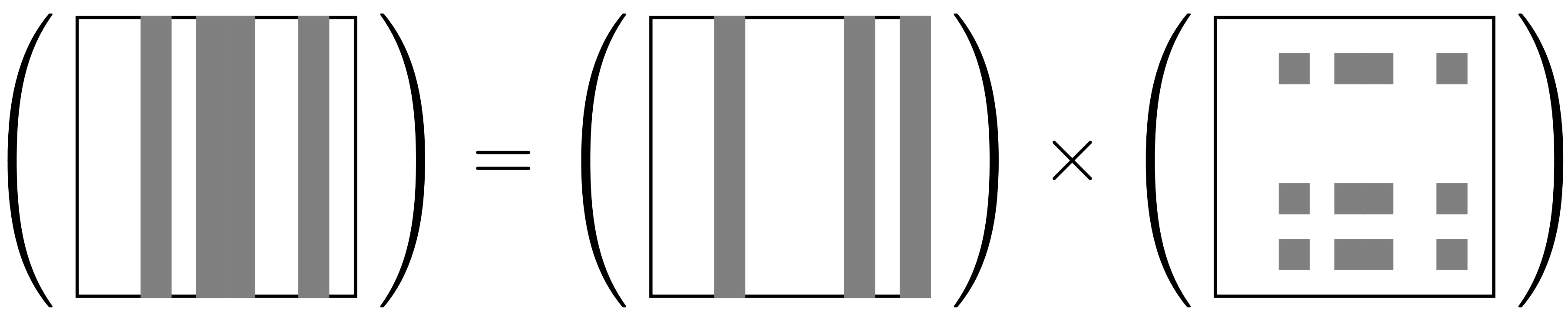
$f_{\ell-1}: G/H \to \mathbb{C}$ $\qquad$ $f_\ell: G \to \mathbb{C}$



$\widehat{f * g}(\rho) \qquad \widehat{f{\uparrow}^G}(\rho) \qquad \widehat{g{\uparrow}^G}(\rho)$

Case 2:

$f_{\ell-1} : G/H \to \mathbb{C}$ $\qquad$ $f_{\ell} : G/K \to \mathbb{C}$



$\widehat{f * g}(\rho) \qquad \widehat{f\!\uparrow^{G}}(\rho) \qquad \widehat{g\!\uparrow^{G}}(\rho)$

3. The nonlinear part

Nonlinearities

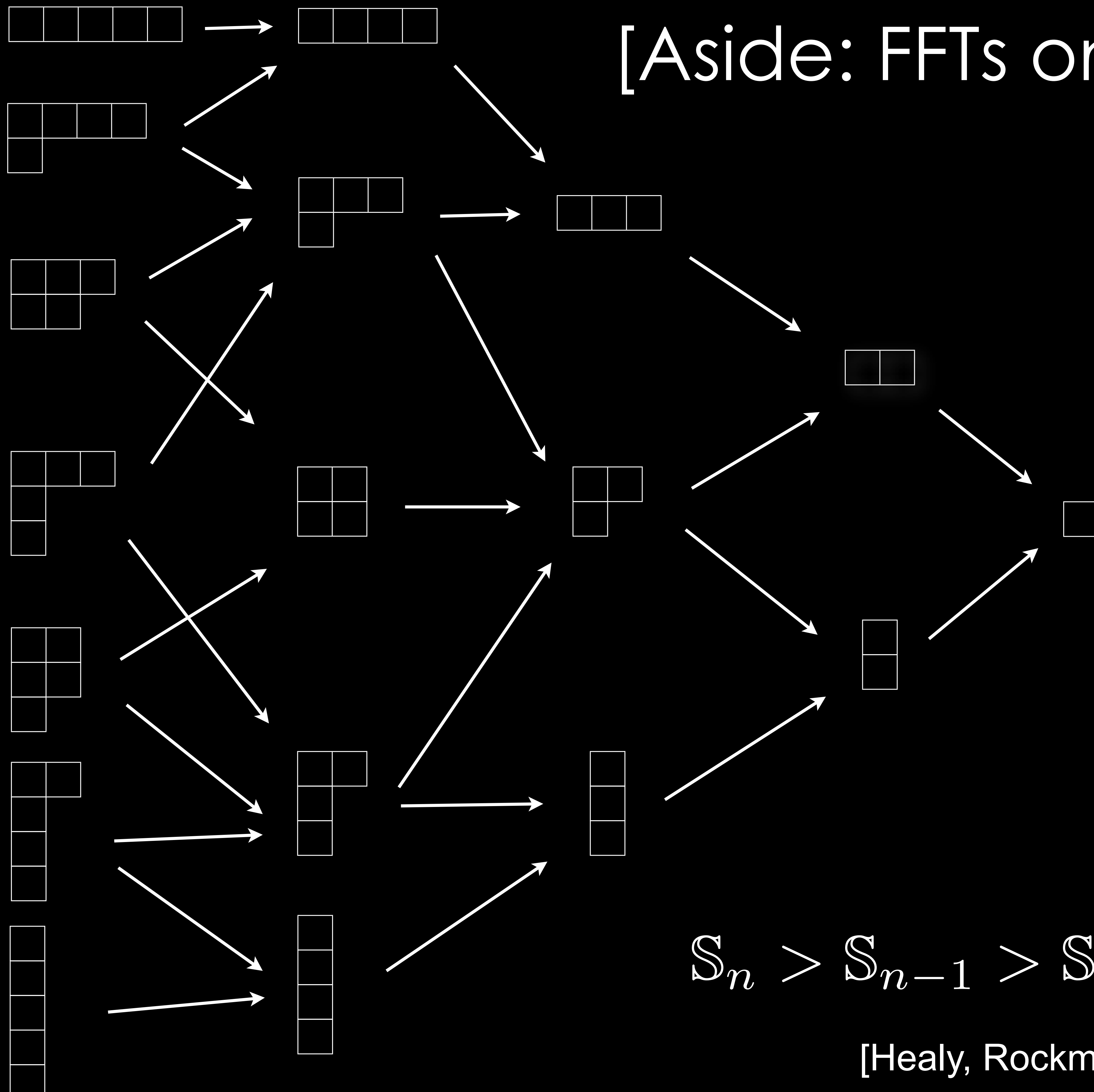
Pointwise:

$$h(u) = \mathrm{ReLU}(f(u)) \qquad h(u-t) = \mathrm{ReLU}(f(u-t))$$

In Fourier space:

$$h(u) = (f(u))^2 \qquad \widehat{h}(k) = \int_{k'} \widehat{f}(k-k')\,\widehat{f}(k')\,dk'$$

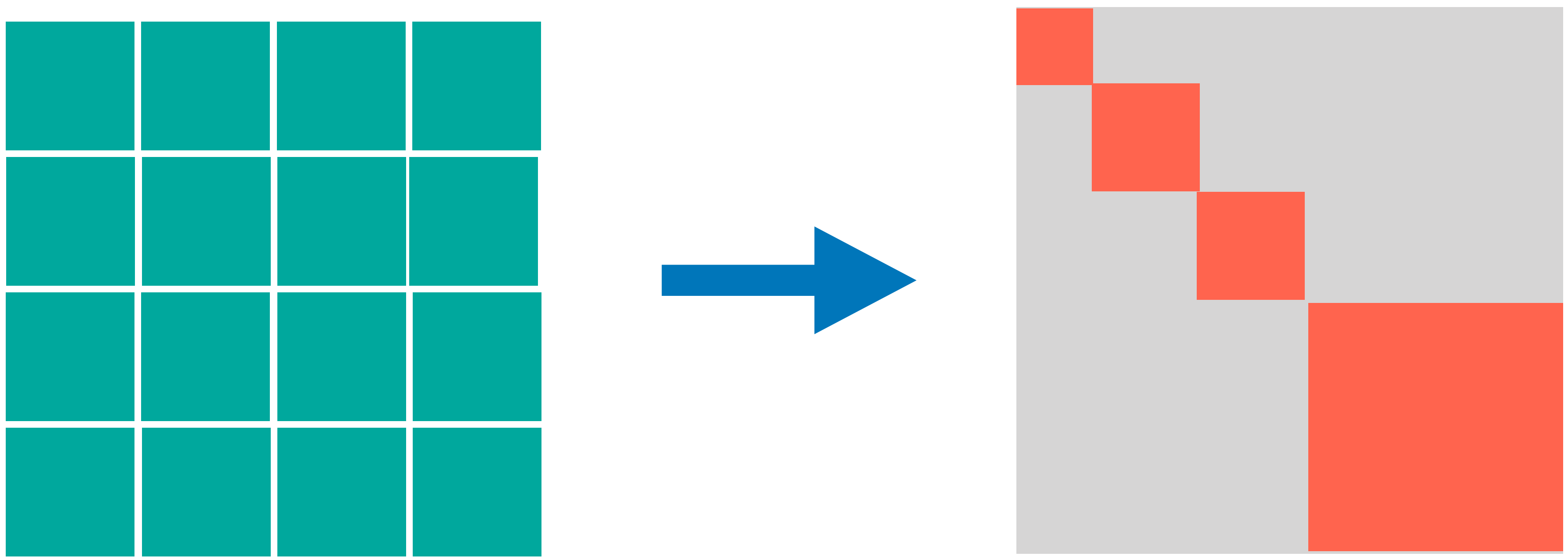
[Aside: FFTs on groups]



$$\mathbb{S}_n > \mathbb{S}_{n-1} > \mathbb{S}_{n-2} > \ldots \mathbb{S}_1$$

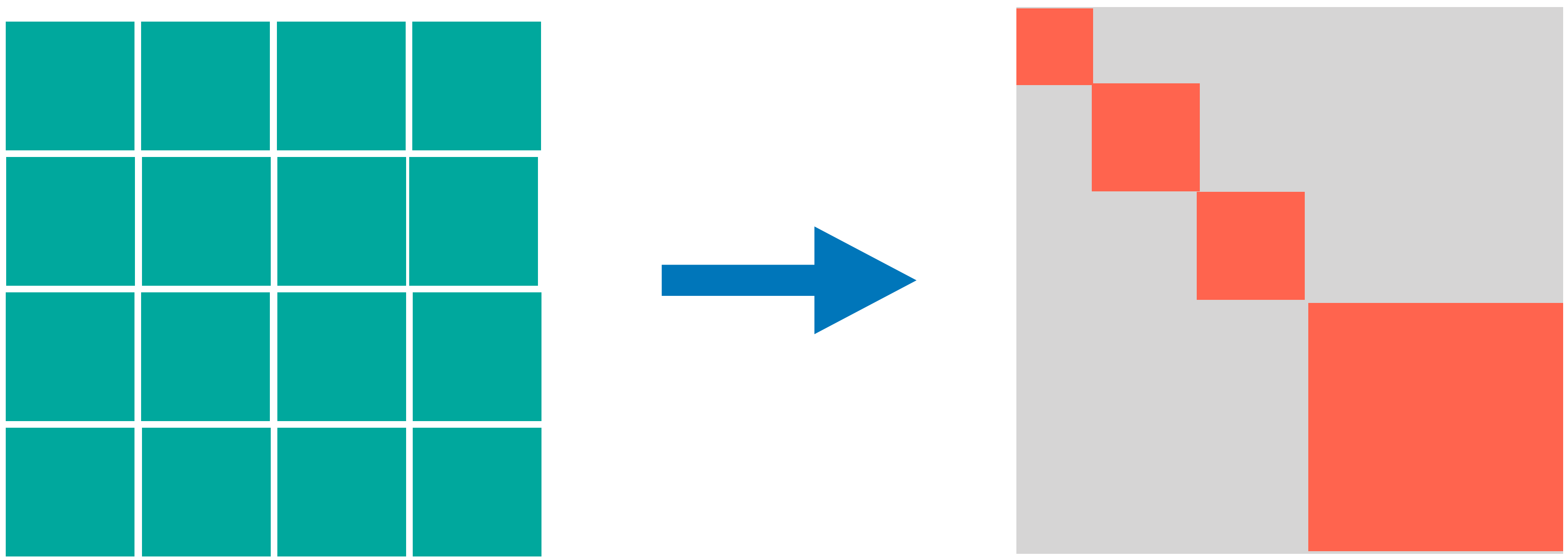
[Healy, Rockmore & Moore, 1996]

Clebsch-Gordan nonlinearities



$$\rho_1(g) \otimes \rho_2(g) = C_{\rho_1,\rho_2} \Big[\bigoplus_{\rho} \bigoplus_{1}^{\kappa(\rho)} \rho(g) \Big] C^{\dagger}_{\rho_1,\rho_2}$$

Clebsch-Gordan nonlinearities



$$\widehat{f}_1(\rho_1) \otimes \widehat{f}_2(\rho_2) = C_{\rho_1,\rho_2} \Big[\bigoplus_{\rho} \bigoplus_{1}^{\kappa(\rho)} \widehat{h}(\rho) \Big] C_{\rho_1,\rho_2}^{\dagger}$$

The SO(3) case

$$D_{\ell_1}(\alpha,\beta,\gamma)\otimes D_{\ell_2}(\alpha,\beta,\gamma) = C_{\ell_1,\ell_2}\left[\bigoplus_{\ell=|\ell_1-\ell_2|}^{\ell_1+\ell_2} D_\ell(\alpha,\beta,\gamma)\right]C^{\dagger}_{\ell_1,\ell_2}$$

Note: A square-root sign is to be understood over *every* coefficient, *e.g.*, for $-8/15$ read $-\sqrt{8/15}$.

Notation:

		J	J	...
		M	M	...
m_1	m_2	Coefficients		
m_1	m_2			
⋮	⋮			

1/2×1/2

m_1	m_2	J=1, M=+1
+1/2	+1/2	1

m_1	m_2	J=1, M=0	J=0, M=0
+1/2	−1/2	1/2	1/2
−1/2	+1/2	1/2	−1/2

m_1	m_2	J=1, M=−1
−1/2	−1/2	1

$$Y_1^0 = \sqrt{\frac{3}{4\pi}}\cos\theta$$

$$Y_1^1 = -\sqrt{\frac{3}{8\pi}}\,\sin\theta\, e^{i\phi}$$

$$Y_2^0 = \sqrt{\frac{5}{4\pi}}\Big(\frac{3}{2}\cos^2\theta - \frac{1}{2}\Big)$$

$$Y_2^1 = -\sqrt{\frac{15}{8\pi}}\,\sin\theta\cos\theta\, e^{i\phi}$$

$$Y_2^2 = \frac{1}{4}\sqrt{\frac{15}{2\pi}}\,\sin^2\theta\, e^{2i\phi}$$

2×1/2

m_1	m_2	J=5/2, M=+5/2
+2	+1/2	1

m_1	m_2	J=5/2, M=+3/2	J=3/2, M=+3/2
+2	−1/2	1/5	4/5
+1	+1/2	4/5	−1/5

m_1	m_2	J=5/2, M=+1/2	J=3/2, M=+1/2
+1	−1/2	2/5	3/5
0	+1/2	3/5	−2/5

m_1	m_2	J=5/2, M=−1/2	J=3/2, M=−1/2
0	−1/2	3/5	2/5
−1	+1/2	2/5	−3/5

m_1	m_2	J=5/2, M=−3/2	J=3/2, M=−3/2
−1	−1/2	4/5	1/5
−2	+1/2	1/5	−4/5

m_1	m_2	J=5/2, M=−5/2
−2	−1/2	1

1×1/2

m_1	m_2	J=3/2, M=+3/2
+1	+1/2	1

m_1	m_2	J=3/2, M=+1/2	J=1/2, M=+1/2
+1	−1/2	1/3	2/3
0	+1/2	2/3	−1/3

m_1	m_2	J=3/2, M=−1/2	J=1/2, M=−1/2
0	−1/2	2/3	1/3
−1	+1/2	1/3	−2/3

m_1	m_2	J=3/2, M=−3/2
−1	−1/2	1

3/2×1/2

m_1	m_2	J=2, M=+2
+3/2	+1/2	1

m_1	m_2	J=2, M=+1	J=1, M=+1
+3/2	−1/2	1/4	3/4
+1/2	+1/2	3/4	−1/4

m_1	m_2	J=2, M=0	J=1, M=0
+1/2	−1/2	1/2	1/2
−1/2	+1/2	1/2	−1/2

m_1	m_2	J=2, M=−1	J=1, M=−1
−1/2	−1/2	3/4	1/4
−3/2	+1/2	1/4	−3/4

m_1	m_2	J=2, M=−2
−3/2	−1/2	1

2×1

m_1	m_2	J=3, M=+3
+2	+1	1

m_1	m_2	J=3, M=+2	J=2, M=+2
+2	0	1/3	2/3
+1	+1	2/3	−1/3

m_1	m_2	J=3, M=+1	J=2, M=+1	J=1, M=+1
+2	−1	1/15	1/3	3/5
+1	0	8/15	1/6	−3/10
0	+1	2/5	−1/2	1/10

m_1	m_2	J=3, M=0	J=2, M=0	J=1, M=0
+1	−1	1/5	1/2	3/10
0	0	3/5	0	−2/5
−1	+1	1/5	−1/2	3/10

m_1	m_2	J=3, M=−1	J=2, M=−1	J=1, M=−1
0	−1	2/5	1/2	1/10
−1	0	8/15	−1/6	−3/10
−2	+1	1/15	−1/3	3/5

m_1	m_2	J=3, M=−2	J=2, M=−2
−1	−1	2/3	1/3
−2	0	1/3	−2/3

m_1	m_2	J=3, M=−3
−2	−1	1

3/2×1

m_1	m_2	J=5/2, M=+5/2
+3/2	+1	1

m_1	m_2	J=5/2, M=+3/2	J=3/2, M=+3/2
+3/2	0	2/5	3/5
+1/2	+1	3/5	−2/5

m_1	m_2	J=5/2, M=+1/2	J=3/2, M=+1/2	J=1/2, M=+1/2
+3/2	−1	1/10	2/5	1/2
+1/2	0	3/5	1/15	−1/3
−1/2	+1	3/10	−8/15	1/6

m_1	m_2	J=5/2, M=−1/2	J=3/2, M=−1/2	J=1/2, M=−1/2
+1/2	−1	3/10	8/15	1/6
−1/2	0	3/5	−1/15	−1/3
−3/2	+1	1/10	−2/5	1/2

m_1	m_2	J=5/2, M=−3/2	J=3/2, M=−3/2
−1/2	−1	3/5	2/5
−3/2	0	2/5	−3/5

m_1	m_2	J=5/2, M=−5/2
−3/2	−1	1

1×1

m_1	m_2	J=2, M=+2
+1	+1	1

m_1	m_2	J=2, M=+1	J=1, M=+1
+1	0	1/2	1/2
0	+1	1/2	−1/2

m_1	m_2	J=2, M=0	J=1, M=0	J=0, M=0
+1	−1	1/6	1/2	1/3
0	0	2/3	0	−1/3
−1	+1	1/6	−1/2	1/3

m_1	m_2	J=2, M=−1	J=1, M=−1
0	−1	1/2	1/2
−1	0	1/2	−1/2

m_1	m_2	J=2, M=−2
−1	−1	1

$$Y_\ell^{-m} = (-1)^m Y_\ell^{m*}$$

$$d^{\,\ell}_{m,0} = \sqrt{\frac{4\pi}{2\ell+1}}\, Y_\ell^m\, e^{-im\phi}$$

$$\langle j_1 j_2 m_1 m_2 | j_1 j_2 J M\rangle = (-1)^{J-j_1-j_2}\langle j_2 j_1 m_2 m_1 | j_2 j_1 J M\rangle$$

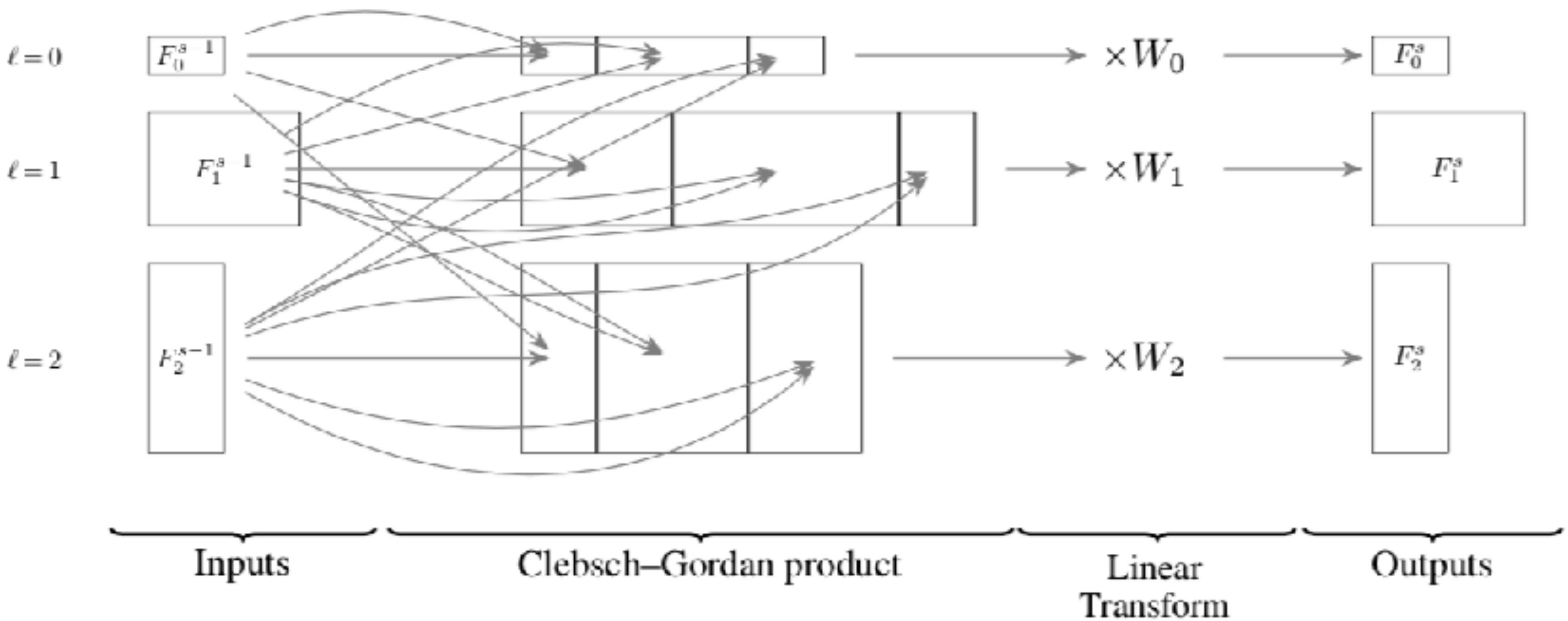
The Clebsch–Gordan coefficients are the solutions to

$$|j_1, j_2; J, M\rangle = \sum_{m_1=-j_1}^{j_1} \sum_{m_2=-j_2}^{j_2} |j_1, m_1; j_2, m_2\rangle \langle j_1, j_2; m_1, m_2 \mid j_1, j_2; J, M\rangle$$

Explicitly:

$$\begin{aligned} &\langle j_1, j_2; m_1, m_2 \mid j_1, j_2; J, M\rangle \\ &= \delta_{M, m_1+m_2} \sqrt{\frac{(2J+1)(J+j_1-j_2)!(J-j_1+j_2)!(j_1+j_2-J)!}{(j_1+j_2+J+1)!}} \times \\ &\sqrt{(J+M)!(J-M)!(j_1-m_1)!(j_1+m_1)!(j_2-m_2)!(j_2+m_2)!} \times \\ &\sum_k \frac{(-1)^k}{k!(j_1+j_2-J-k)!(j_1-m_1-k)!(j_2+m_2-k)!(J-j_2+m_1+k)!(J-j_1-m_2+k)!}. \end{aligned}$$

The summation is extended over all integer k for which the argument of every factorial is nonnegative.[4]



c.f. [Cohen, Geiger, Köhler & Welling, 2018] [Esteves, Allen-Blanchette, Makadia, Daniilidis, 2017]

4. Dirty details

Theory

Experiment

Implementation

FastCG

There are 5 basic types in FastCG:

- **`Rscalar<TYPE>`**
- **`Cscalar<TYPE>`**
- **`Ctensor<TYPE>`**
- **`SO3part<TYPE>`**
- **`SO3vec<TYPE>`**

`SO3part` and **`SO3vec`** are full fledged arithmetic types with a lot of optimization behind them, not just collections of tensors.

```
int main(int argc, char** argv){

  SO3type tau({1,1,1});

  SO3vec<float> v(tau, fill::gaussian);

  cout<<v<<endl;

}
```

```
Part l=0:
  [ (-1.23974,-0.407472) ]
Part l=1:
  [ (1.61201,0.399771) (1.3828,0.0523187) (-0.904146,1.87065) ]
Part l=2:
  [ (-1.66043,-0.688081) (0.0757219,1.47339) (0.097221,-0.89237) (-0.228782,1.16493)
(0.584898,-0.660558) ]
```

```
SO3vec<float> u({1,1},fill::gaussian);
SO3vec<float> v({1,1},fill::gaussian);

SO3vecVar<float> w=u+v;
cout<<"w="<<w<<endl;

SO3vecVar<float> cg=CGproduct(u,v);
cout<<"cg="<<cg<<endl;
```

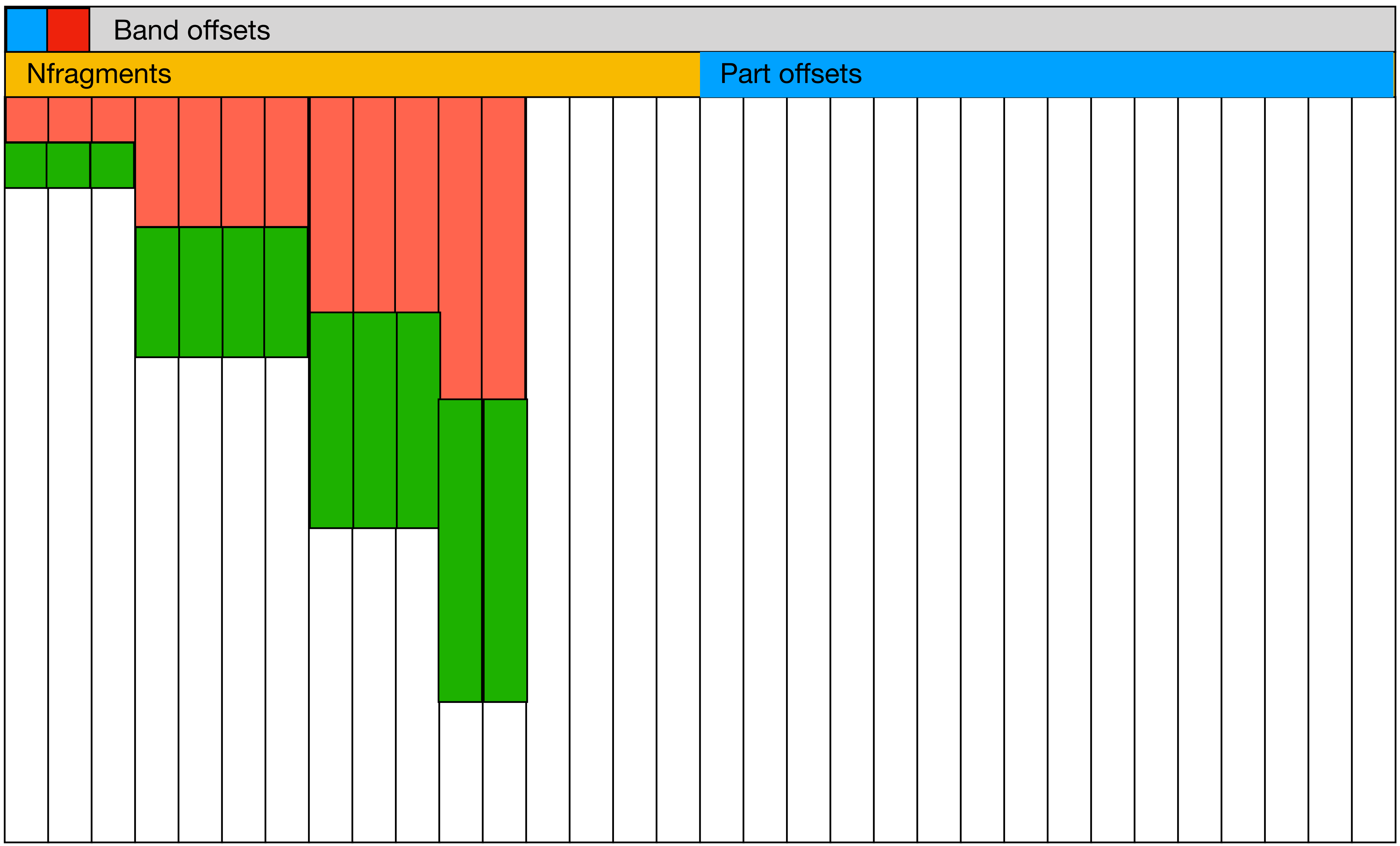
```
Part l=0:
  [ (1.77812,1.52962) ]
  [ (-2.21719,1.05355) ]
Part l=1:
  [ (0.506489,-1.85747) (-0.484144,1.06669) (0.758306,-1.35099) ]
  [ (-2.40155,-1.77298) (-2.26004,-1.03835) (2.78843,-2.48396) ]
  [ (-0.343542,2.43316) (-1.40725,-2.10501) (1.38502,-0.431477) ]
Part l=2:
  [ (-0.466953,2.40539) (0.382606,0.453766) (-1.34596,-0.7601) (0.851431,1.82971) (-1.97233,-1.4812
```

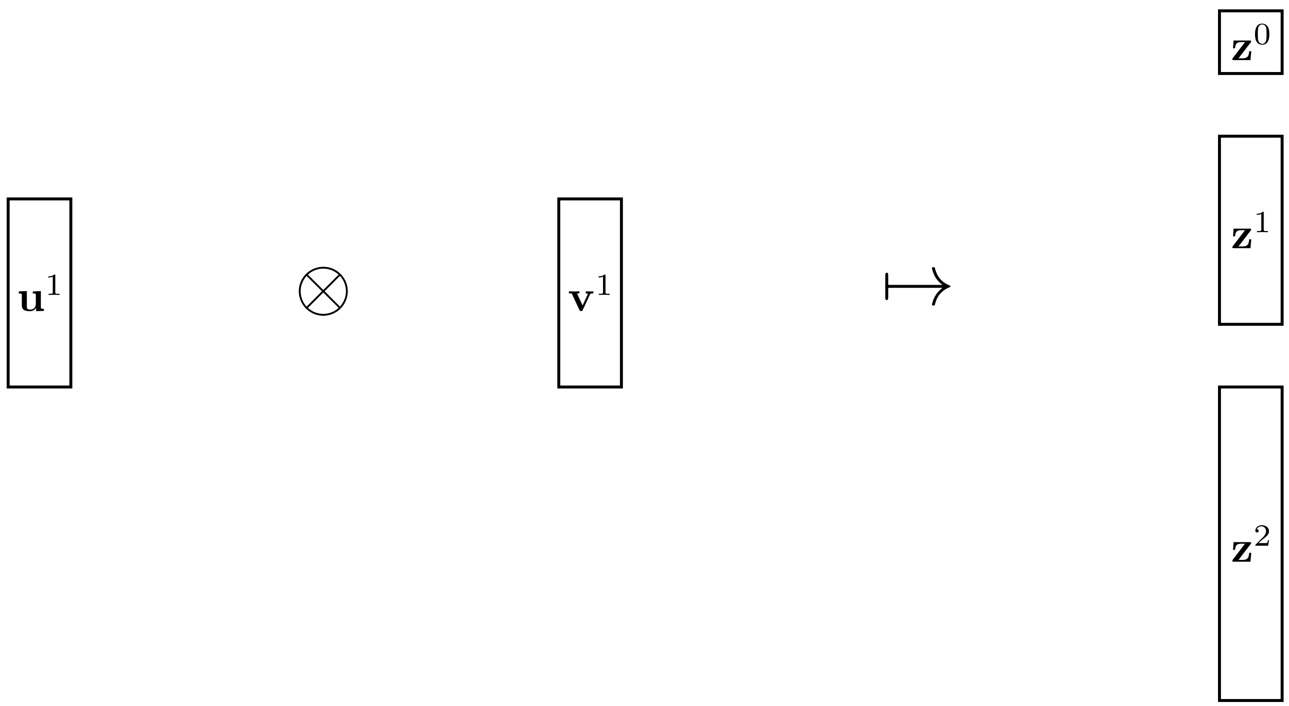
```
class MyNet: public NeuralNet<SO3vec<float>>{
public:

  MyNet(){

    Input<SO3vec<float>> v1;
    Input<SO3vec<float>> v2;
    SO3vecVar<float> w=v1+v2;

    SO3vecVar<float> u=CGproduct(w,v1+w);

    output[0]=u;

  }

};
```

```
SO3vec<float> a({2,3},fill::gaussian);
SO3vec<float> b({1,0,1},fill::gaussian);

MyNet<SO3vec<float>> my_net;
cout<<my_net(a,b)<<endl;

SO3vec<float> g({2,6,4,3},fill::gaussian);
auto pack=my_net.backward(g);
cout<<pack[0]<<endl;
```

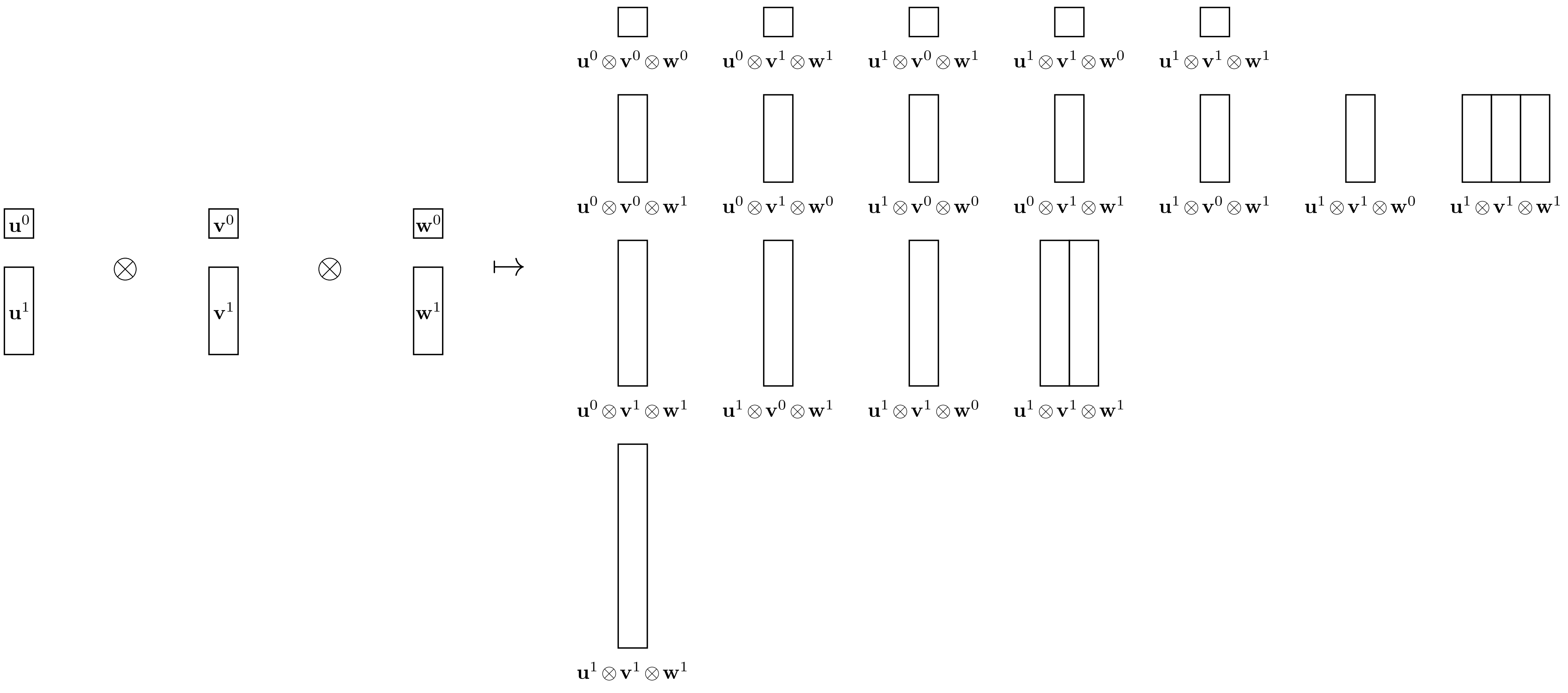
```
class MyNet: public NeuralNet<SO3vec<float>>{
public:

  MyNet(){

    Input<SO3vec<float>> v1;
    Input<SO3vec<float>> v2;
    SO3vecVar<float> w=v1+v2;

    SO3vecVar<float> u=CGproduct(w,v1+w);

    output[0]=u;

  }

};
```

GPU friendly storage

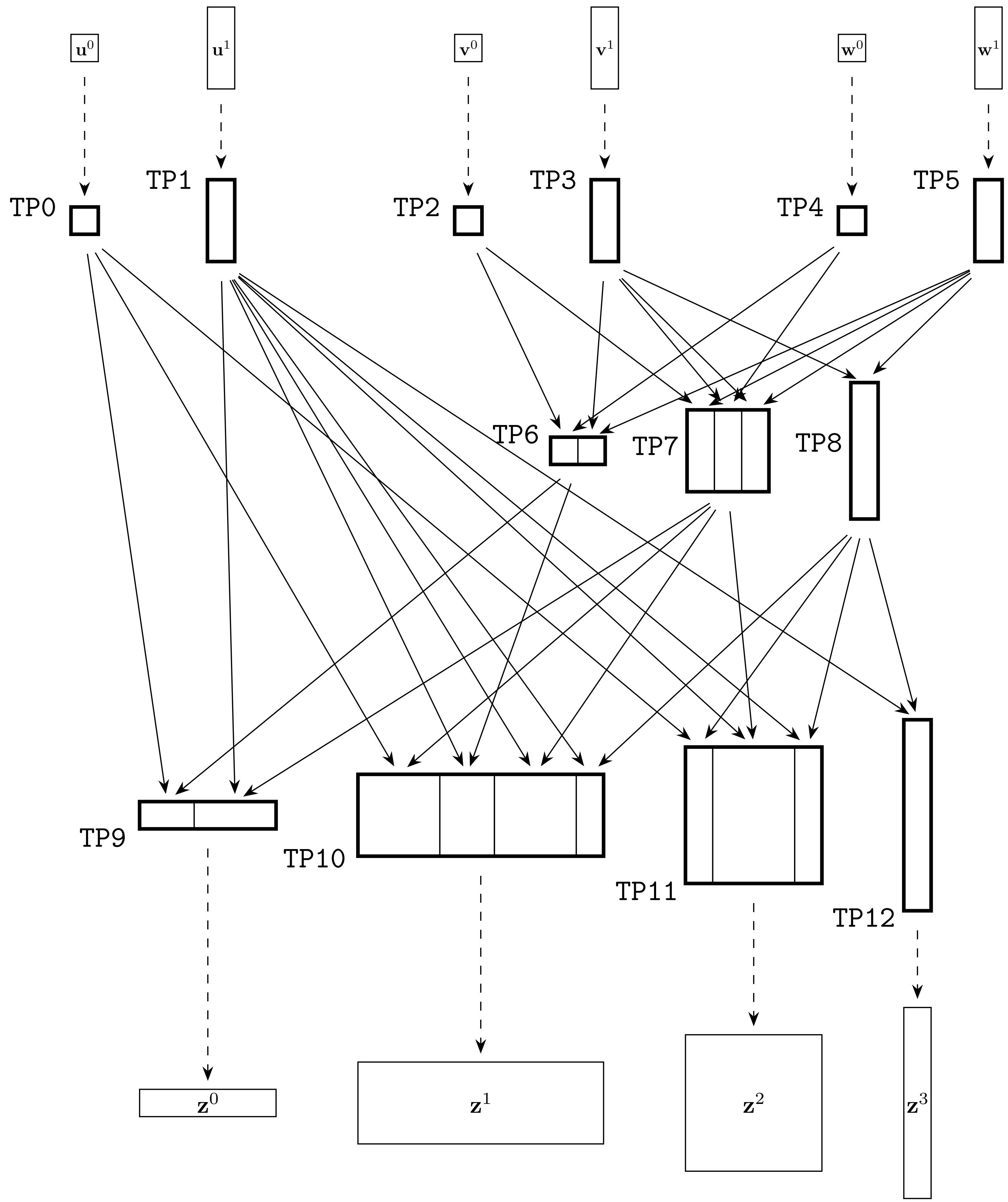




$\mathbf{u}^1$
$\otimes$
$\mathbf{v}^1$
$\mapsto$
$\mathbf{z}^0$
$\mathbf{z}^1$
$\mathbf{z}^2$

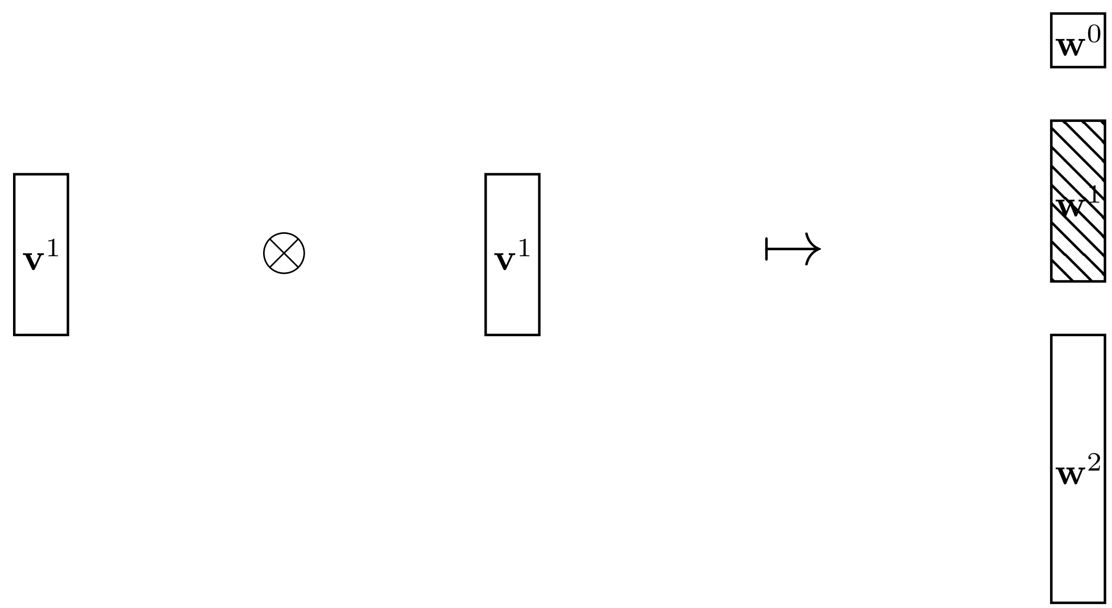


$\mathbf{u}^0$
$\mathbf{u}^1$
$\mathbf{v}^0$
$\mathbf{v}^1$
$\mathbf{w}^0$
$\mathbf{w}^1$
$\mathbf{u}^0 \otimes \mathbf{v}^0 \otimes \mathbf{w}^0$
$\mathbf{u}^0 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^0 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^0$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$
$\mathbf{u}^0 \otimes \mathbf{v}^0 \otimes \mathbf{w}^1$
$\mathbf{u}^0 \otimes \mathbf{v}^1 \otimes \mathbf{w}^0$
$\mathbf{u}^1 \otimes \mathbf{v}^0 \otimes \mathbf{w}^0$
$\mathbf{u}^0 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^0 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^0$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$
$\mathbf{u}^0 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^0 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^0$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$
$\mathbf{u}^1 \otimes \mathbf{v}^1 \otimes \mathbf{w}^1$

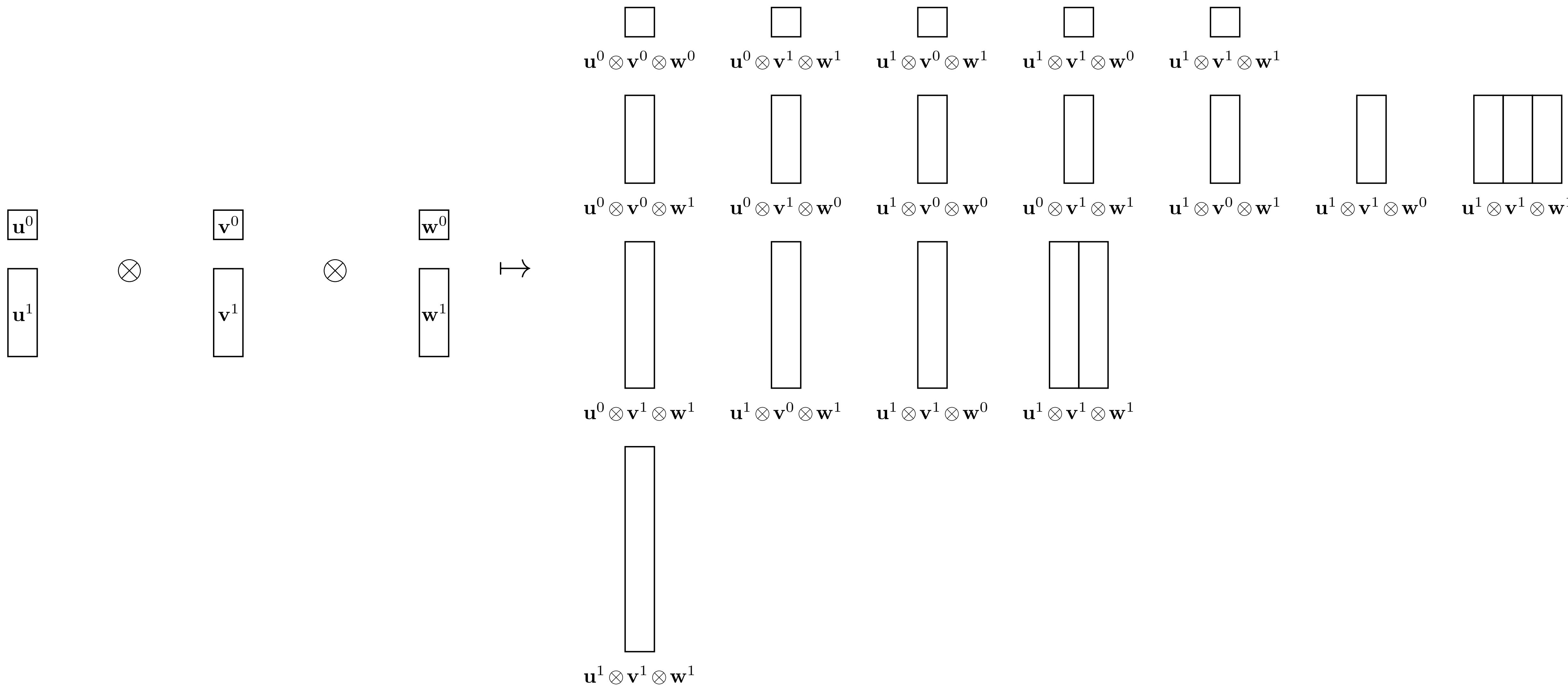


u0
u1
v0
v1
w0
w1
TP0
TP1
TP2
TP3
TP4
TP5
TP6
TP7
TP8
TP9
TP10
TP11
TP12
z0
z1
z2
z3

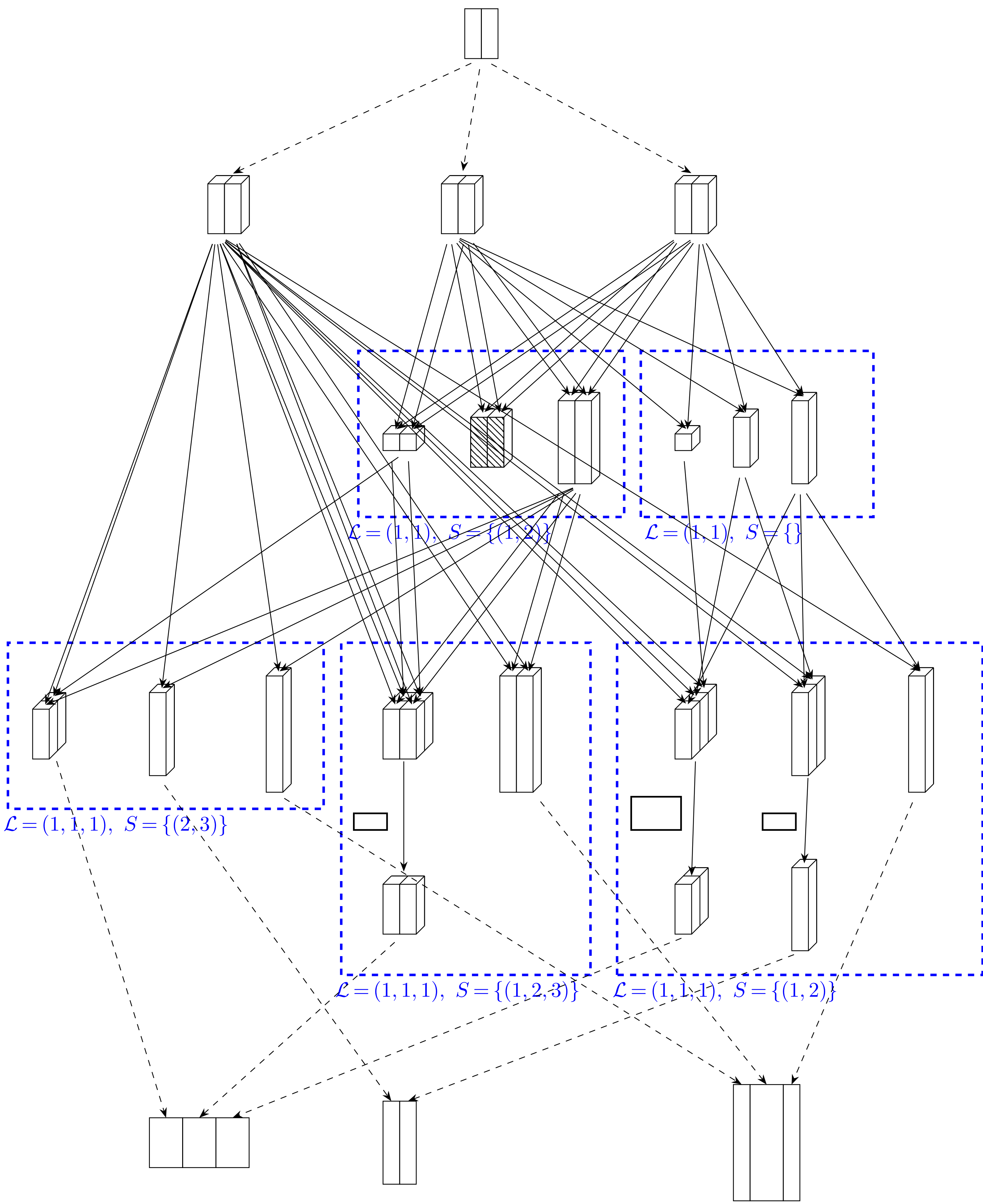
```
TPprogram  CGproduct(){
  TPpart0 (l=0)[0m (n=1){
    input(0,0);
  }
  TPpart1 (l=1) (n=1){
    input(0,1);
  }
  TPpart2 (l=0) (n=1){
    input(1,0);
  }
  TPpart3 (l=1) (n=1){
    input(1,1);
  }
  TPpart4 (l=0) (n=1){
    input(2,0);
  }
  TPpart5 (l=1) (n=1){
    input(2,1);
  }
  TPpart6 (l=0) (n=2){
    CG(2,4)[0];
    CG(3,5)[1];
  }
  TPpart7 (l=1) (n=3){
    CG(2,5)[0];
    CG(3,4)[1];
    CG(3,5)[2];
  }
  TPpart8 (l=2) (n=1){
    CG(3,5)[0];
  }
  TPpart9 (l=0) (n=5){
    output(0);
    CG(0,6)[0];
    CG(1,7)[2];
  }
  TPpart10 (l=1) (n=9){
    output(1);
    CG(0,7)[0];
    CG(1,6)[3];
    CG(1,7)[5];
```



$\mathbf{v}^1$
$\otimes$
$\mathbf{v}^1$
$\mapsto$
$\mathbf{w}^0$
$\mathbf{w}^1$
$\mathbf{w}^2$



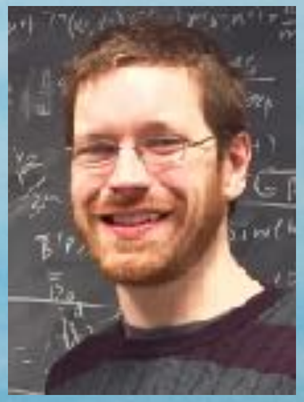
u0
u1
v0
v1
w0
w1
u0 ⊗ v0 ⊗ w0
u0 ⊗ v1 ⊗ w1
u1 ⊗ v0 ⊗ w1
u1 ⊗ v1 ⊗ w0
u1 ⊗ v1 ⊗ w1
u0 ⊗ v0 ⊗ w1
u0 ⊗ v1 ⊗ w0
u1 ⊗ v0 ⊗ w0
u0 ⊗ v1 ⊗ w1
u1 ⊗ v0 ⊗ w1
u1 ⊗ v1 ⊗ w0
u1 ⊗ v1 ⊗ w1
u0 ⊗ v1 ⊗ w1
u1 ⊗ v0 ⊗ w1
u1 ⊗ v1 ⊗ w0
u1 ⊗ v1 ⊗ w1
u1 ⊗ v1 ⊗ w1



$\mathcal{L} = (1, 1),\ S = \{(1, 2)\}$
$\mathcal{L} = (1, 1),\ S = \{\}$
$\mathcal{L} = (1, 1, 1),\ S = \{(2, 3)\}$
$\mathcal{L} = (1, 1, 1),\ S = \{(1, 2, 3)\}$
$\mathcal{L} = (1, 1, 1),\ S = \{(1, 2)\}$

```
SPprogram  TensorPower3(){
  Snode0 () () (l=0) (0x1){
    input(0);
  }
  Snode1 () () (l=1) (2x1){
    input(1);
  }
  Snode2 (1,1) () (l=2) (1x1) weave(0){
    (1,1)[0];
  }
  Snode3 (1,1) () (l=1) (1x1) weave(0){
    (1,1)[0];
  }
  Snode4 (1,1) () (l=0) (1x1) weave(0){
    (1,1)[0];
  }
  Snode5 (1,1) ((0,1)) (l=2) (2x1) weave(1){
    (1,1)[0];
  }
  Snode6 (1,1) ((0,1)) (l=0) (2x1) weave(1){
    (1,1)[0];
  }
  Snode7 (1,1,1) ((1,2)) (l=3) (1x1) weave(2){
    (1,5)[0];
  }
  Snode8 (1,1,1) ((1,2)) (l=2) (1x1) weave(2){
    (1,5)[0];
  }
  Snode9 (1,1,1) ((1,2)) (l=1) (1x2) weave(2){
    (1,5)[0];
    (1,6)[1];
  }
```

5. Cormorant



Brandon Anderson

Hy Trong Son

Physical interactions

Monopole:

$$V_C = -\frac{1}{4\pi\epsilon_0}\frac{q_A q_B}{|\boldsymbol{r}_{AB}|}$$

Dipole:

$$V_{d/d} = \frac{1}{4\pi\epsilon_0}\left[\frac{\boldsymbol{\mu}_A \cdot \boldsymbol{\mu}_B}{|\boldsymbol{r}_{AB}|^3} - 3\,\frac{(\boldsymbol{\mu}_A \cdot \boldsymbol{r}_{AB})(\boldsymbol{\mu}_B \cdot \boldsymbol{r}_{AB})}{|\boldsymbol{r}_{AB}|^5}\right].$$

Quadropole:

$$V_{q/q} = \frac{3}{4}\frac{\vartheta_A \vartheta_B}{4\pi\epsilon_0\,|\boldsymbol{r}_{AB}|^5}\left[1 - 5\cos^{\theta}_A - 5\cos^2\theta_B - 15\cos^2\theta_A\cos^2\theta_B + \right.$$
$$\left. 2(4\cos\theta_A\theta_B - \sin\theta_A\sin\theta_B\cos(\phi_A - \phi_B))^2\right]$$

[Thomas et al: Tensor Field Networks (2018)]

The Cormorant architecture

$$\mathrm{CGLayer}(\{F_i, \mathbf{r}_{ij}\}) = \Big[F_i \oplus \big(F_i \otimes_{\mathrm{cg}} F_i\big) \oplus \Big(\sum_j \Big(\Upsilon_{ij}^{(1)} \oplus \big(F_i \cdot F_j\big)\Upsilon_{ij}^{(3)}\Big) \otimes_{\mathrm{cg}} F_j\Big)\Big] \cdot W'$$

$$\Upsilon^{(n)}(\mathbf{r}_{jj'}) = \bigoplus_{\ell=0}^{\ell_{\max}} \mathcal{F}^{\ell}(r_{ij}) Y^{\ell}(\hat{\mathbf{r}}_{jj'}),$$

[Anderson, Hy & K: Cormorant (2019)]

QM-9 results

	Cormorant	SchNet	NMP	WaveScatt
α (bohr^3)	**0.092**	0.235	**0.092**	0.160
$\Delta\epsilon$ (eV)	**0.060**	**0.063**	0.069	0.118
ϵ_{HOMO} (eV)	**0.036**	0.041	0.043	0.085
ϵ_{LUMO} (eV)	**0.036**	**0.034**	0.038	0.076
μ (D)	0.130	0.033	**0.030**	0.340
C_v (cal/mol K)	**0.031**	0.033	0.040	0.049
R^2 (bohr^2)	0.673	**0.073**	0.180	0.410
U_0 (eV)	0.028	**0.014**	0.020	0.022
ZPVE (meV)	1.982	1.700	**1.500**	2.000

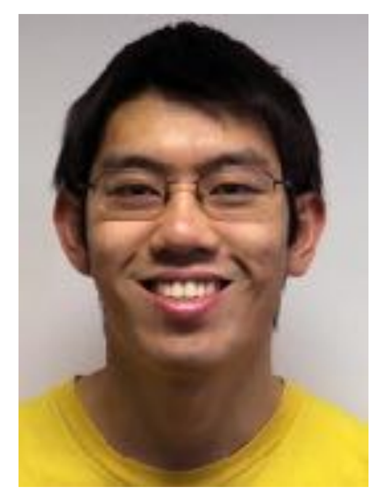
MD-17 results (kcal/mol)

	NBody50k	Deep95k	DTNN50k	SchE50k	SchEF50k	GDML1k	SchE1k	SchEF1k
Aspirin	*0.103	0.201	--	0.250	0.120	0.270	4.200	0.370
Benzene	*0.035	0.065	0.040	0.080	0.070	0.070	1.190	0.080
Ethanol	*0.029	0.055	--	0.070	0.050	0.150	0.930	0.080
Malonaldehyde	*0.056	0.092	0.190	0.130	0.080	0.160	2.030	0.130
Naphthalene	*0.043	0.095	--	0.200	0.110	0.120	3.580	0.160
Salicylic_acid	*0.072	0.106	0.410	0.250	0.100	0.120	3.270	0.200
Toluene	*0.042	0.085	0.180	0.160	0.090	0.120	2.950	0.120
Uracil	*0.045	0.085	--	0.130	0.100	0.110	2.260	0.140

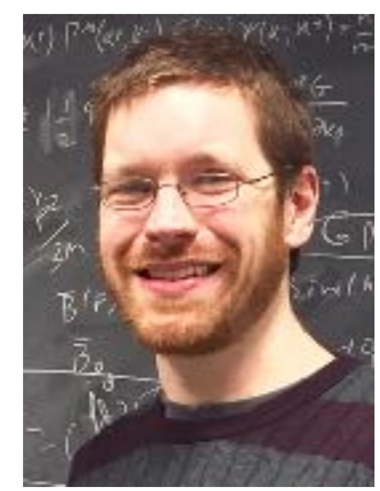
Erik Thiede

Shubhendu Trivedi

Hy Trong Son



Horace Pan



Brandon Anderson