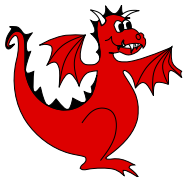


Firedrake

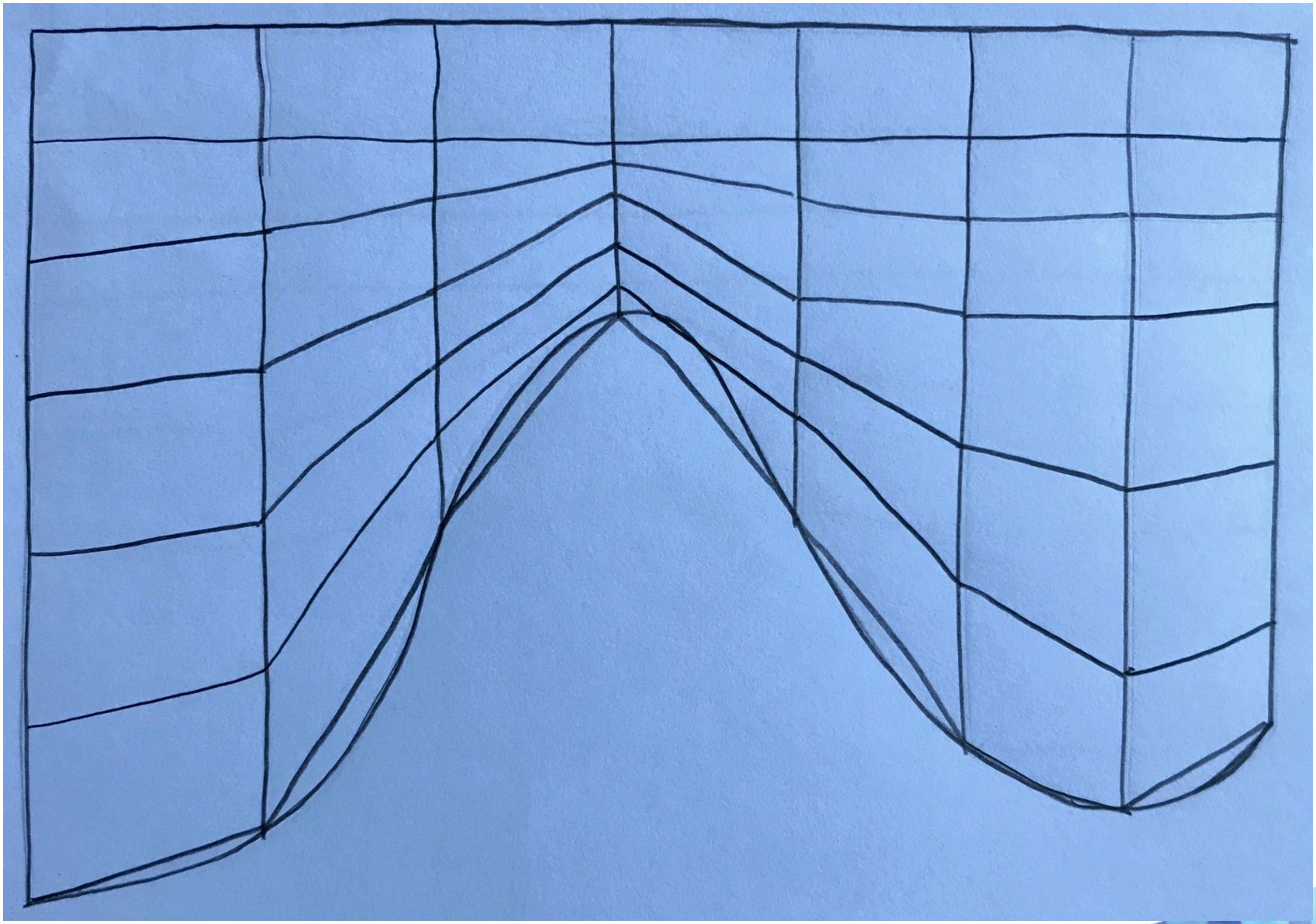
z-layers, oh noes

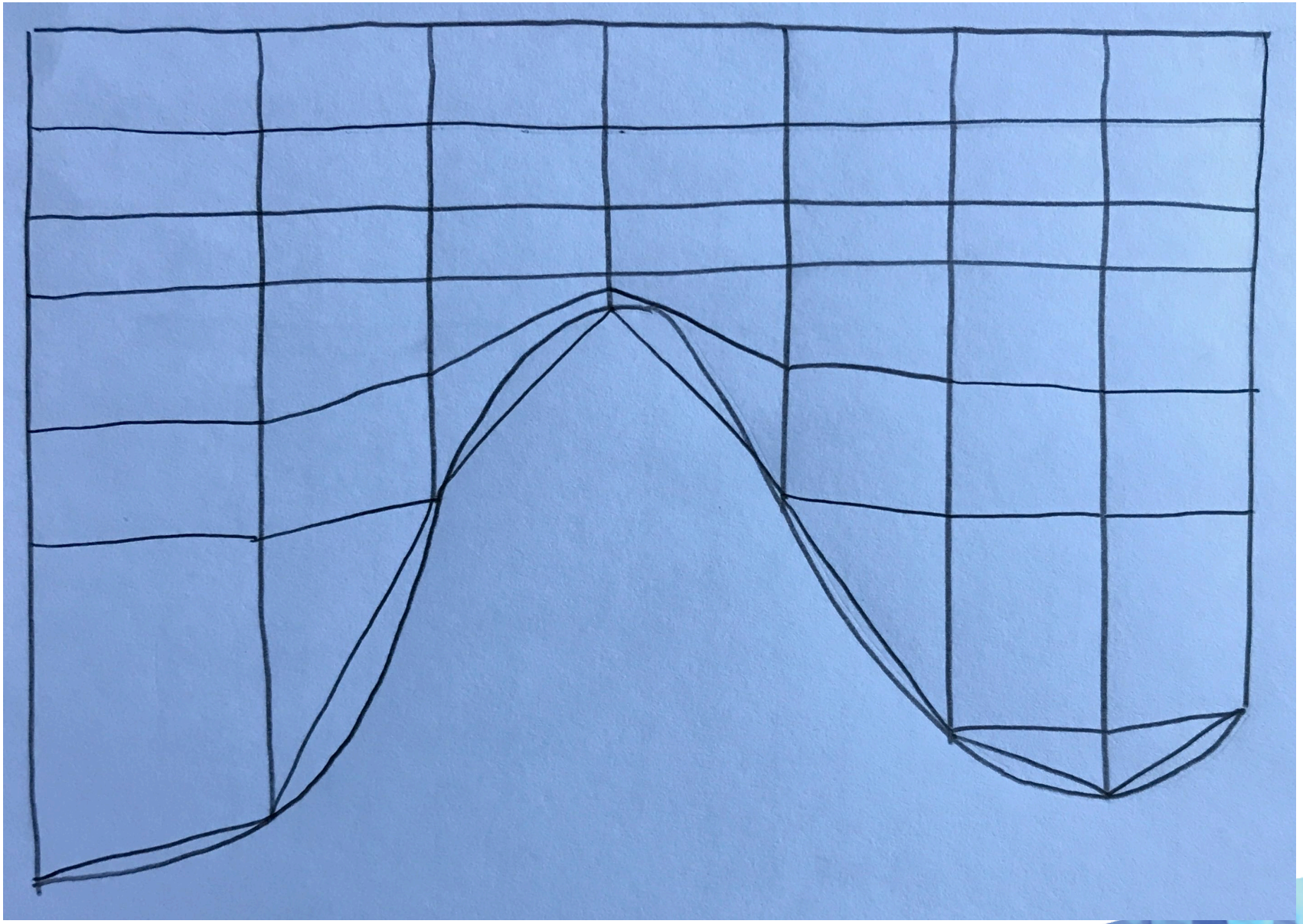
Lawrence Mitchell[1]

18th July 2017

[1]Departments of Computing and Mathematics, Imperial College London

Imperial College London

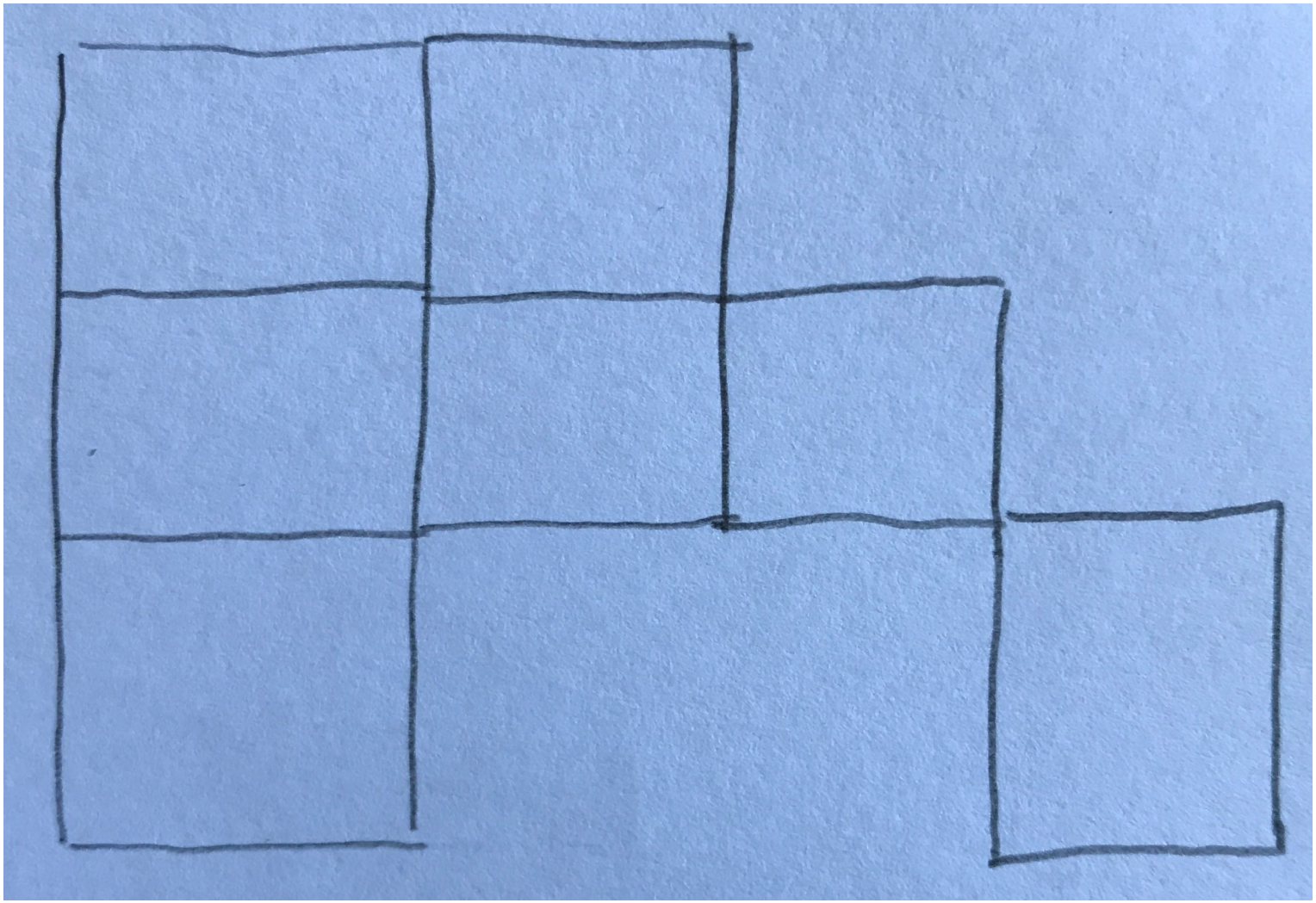




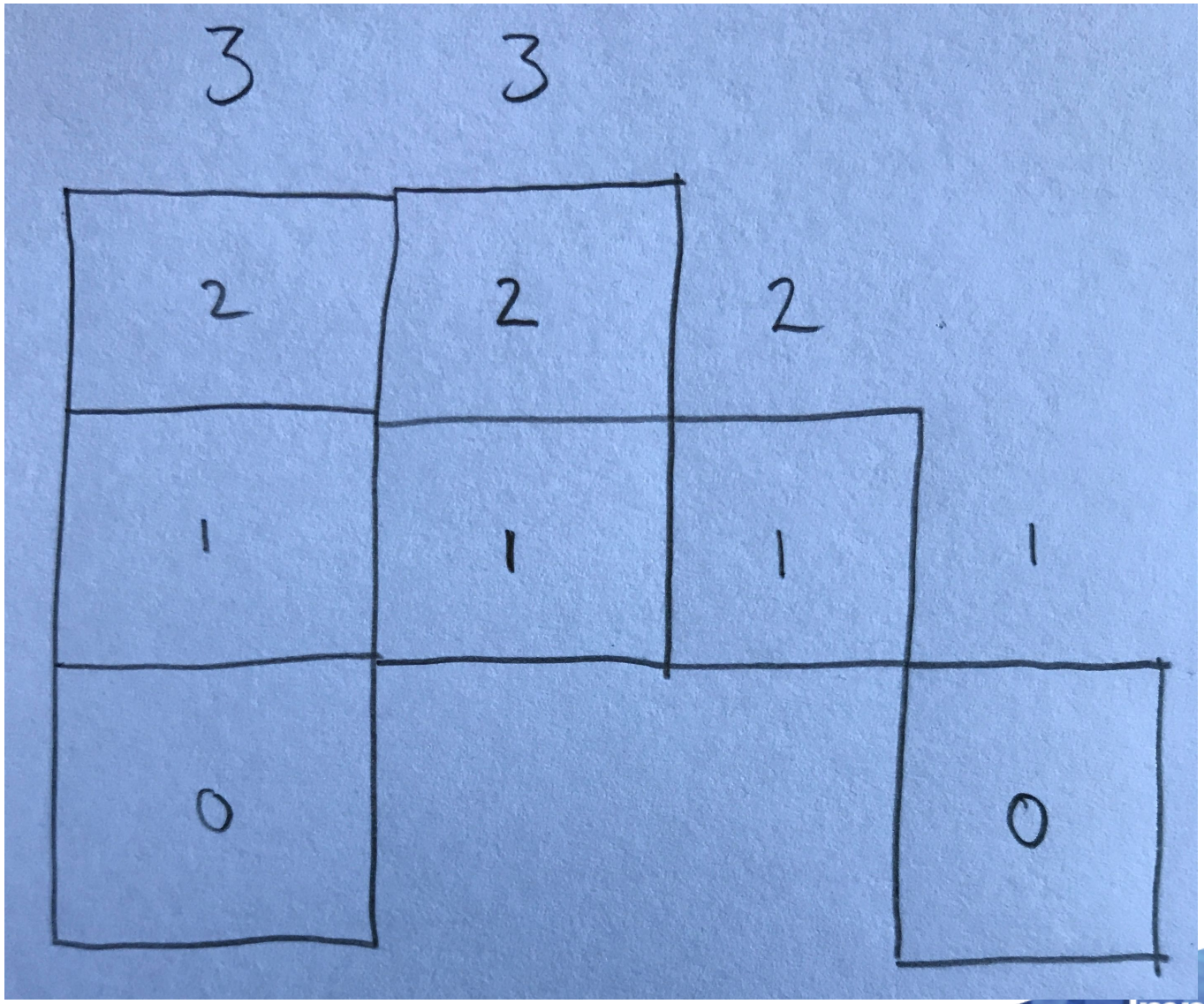
Imperial College London

z-layers

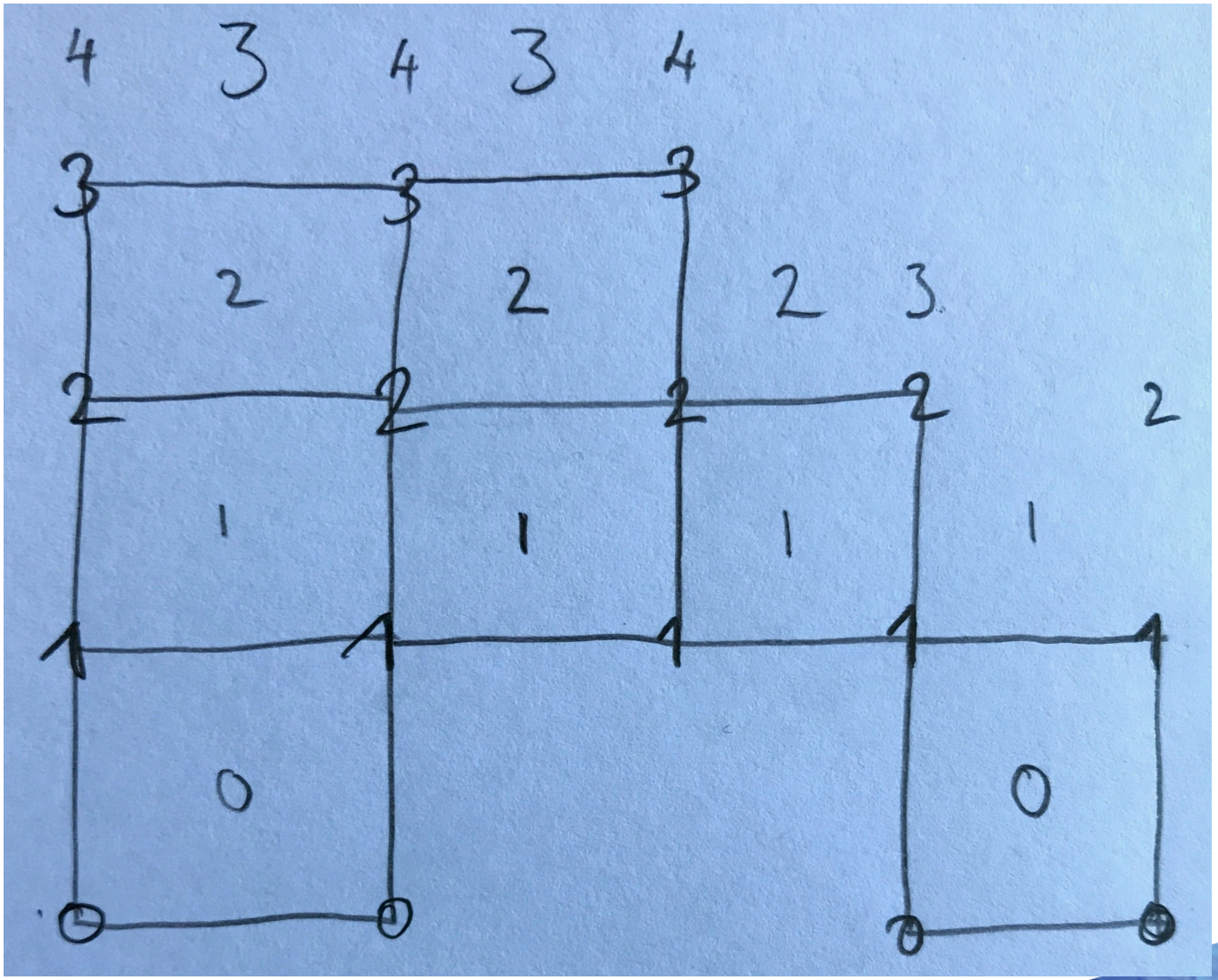
- Around steep topography, might want to use z-layers rather than σ coordinates.
- Mostly the core computational aspects remain unchanged.
- But, the layer number is now *entity*-dependent.
- So there's loads more book-keeping.

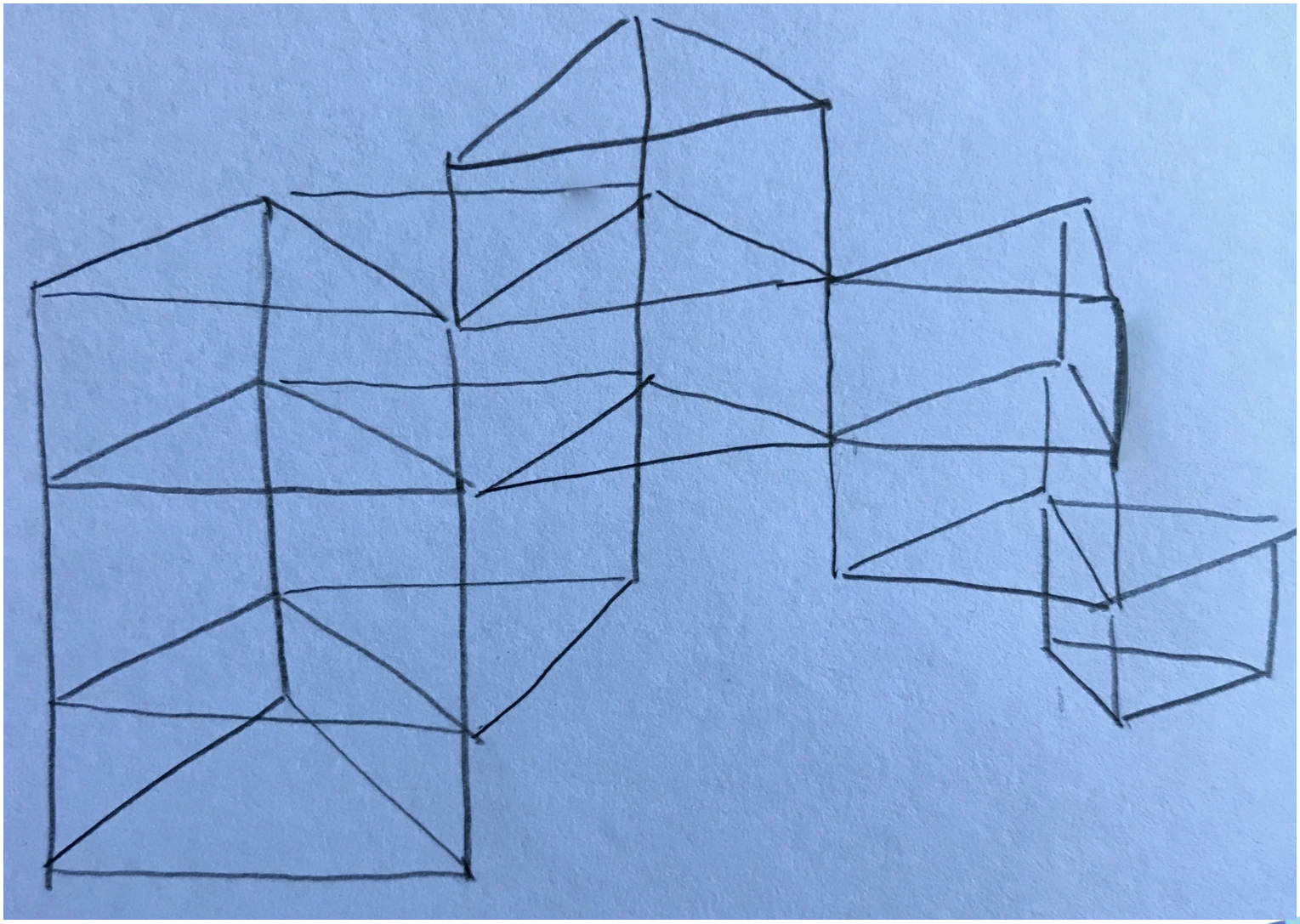


Provide number of cell layers per base cell



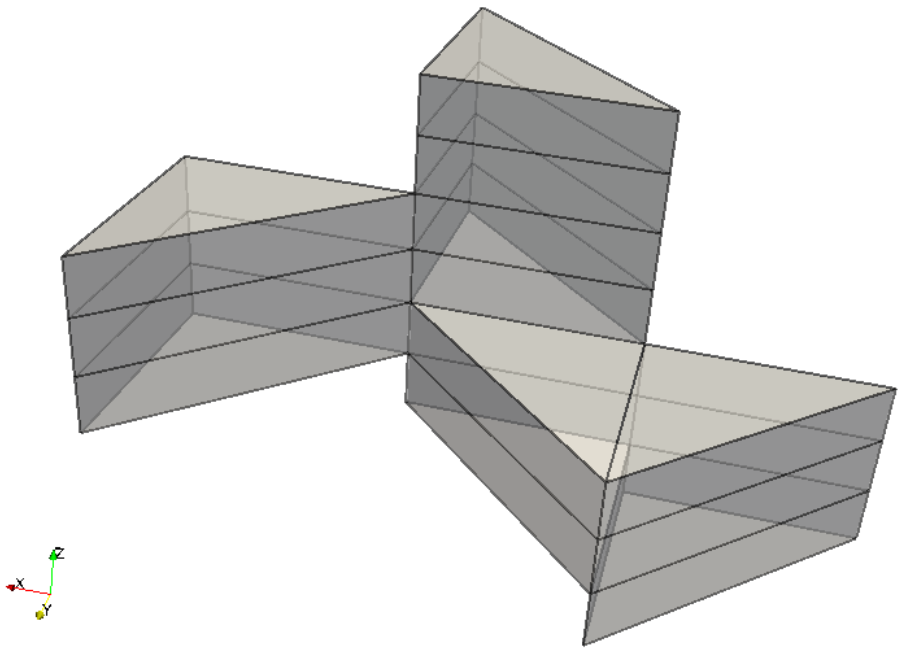
Bootstrap layers for other entities





Mesh construction

- On each base cell, provide:
  1. Start layer (bottom is zero)
  2. Number of cells



```
mesh2d = Mesh(...)
mesh = ExtrudedMesh(mesh2d, [[0, 3],
                             [1, 2],
                             [2, 3],
                             [3, 4]],
                    layer_height=0.25)
```

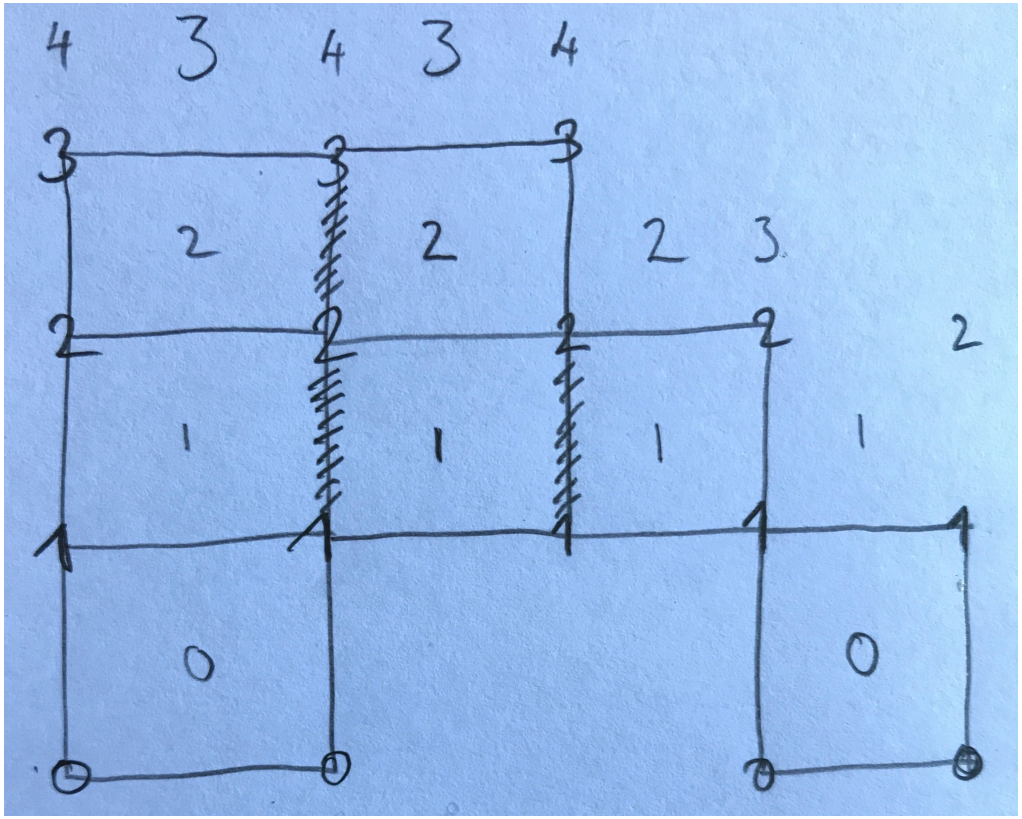
Datatype extension

Iteration sets need *four* values per entry.

- **allocation** First two entries control allocation of dofs. When assigning dofs to base mesh entities, we must consider full column.
- **iteration** Second two control iteration. When iterating over entities, we can't iterate over "exposed" interior facets.
- Need to attach these to *all* base mesh entities.

Datatype extension

Iteration sets need *four* values per entry.




```
interior_facets = ExtrudedSet(2,
                              layers=[[0, 3, 1, 3],
                                      [1, 3, 1, 2]])
```


Imperial College London

Kernel extension

- GungHo *kernel* contains iteration over layers
- So now we need to provide this information at runtime
- Just a directly accessed data array
- This needs to be part of kernel API

```
subroutine old(A, B, nlayers)
   ...
   integer, intent(in) :: nlayers

   do k = 0, nlayers-1
    ...
   end do
end subroutine old
```

```
subroutine new(A, B, lstart, lend)
   ...
   integer, intent(in) :: lstart, lend

   do k = lstart, lend-1
    ...
   end do
end subroutine new
```

Boundary conditions

- As ever, strong conditions are painful
- It is no longer the case that only nodes on the bottom/top cell are killed
- I maintain a bitmask on each cell that marks which topological entities on the cell are exposed
- Then when assembling I can determine which dofs to drop on the floor
- This is easier if you never build sparse matrices: what's the status here?

Firedrake support

- Support is still WIP (interior facets)
- We squash interior facets geometrically, but not topologically.
- Then we never iterate over these facets.
- They could be left exposed, but now need new iteration type.
- An alternate option would be mixed cell shape (ugh!)