? ⇨ gocad ⇨ cubit ⇨ (FEA)

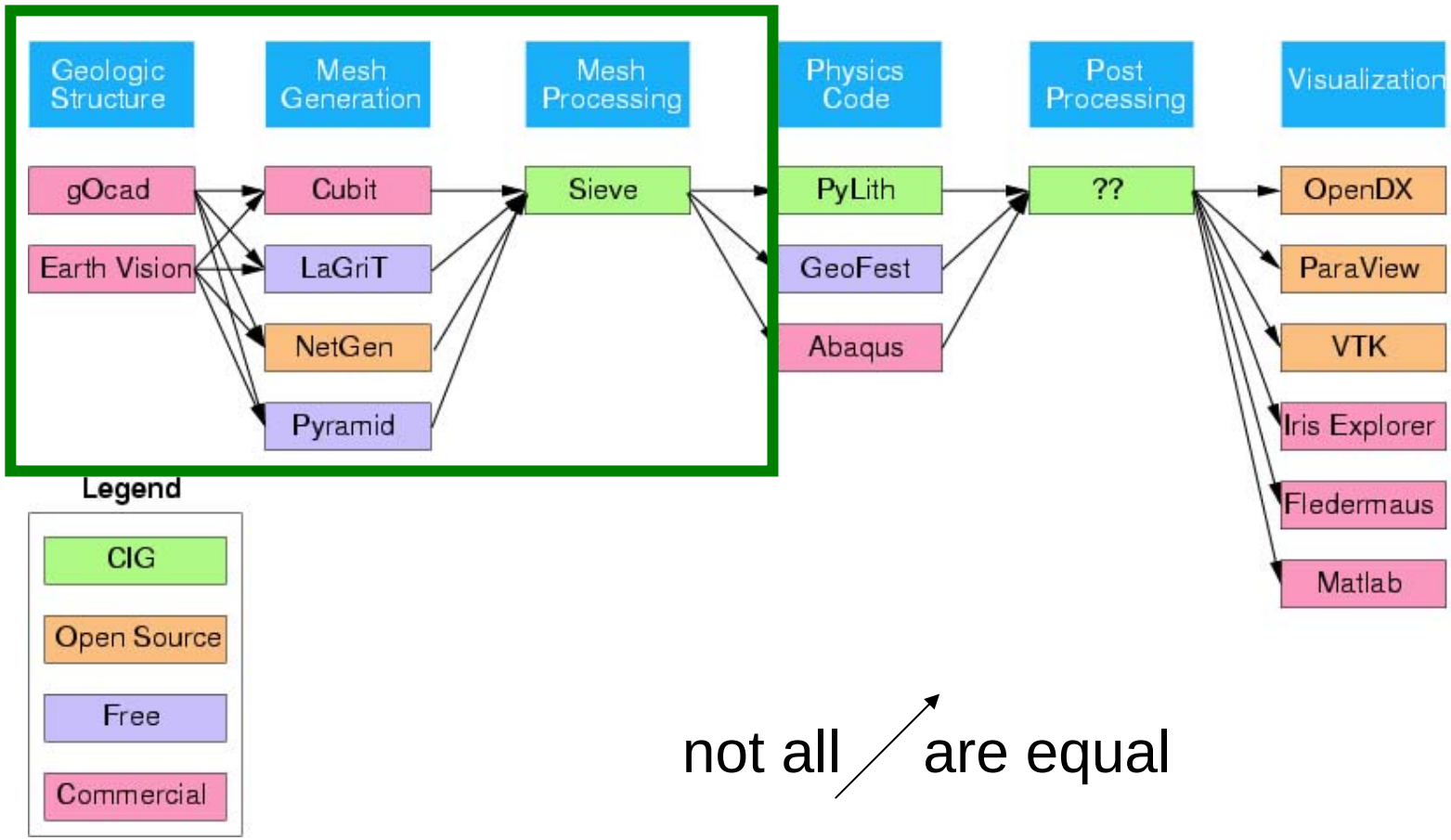
progress report on defining a general work-flow (*geomod*),

and some problems we have encountered along the way…

Eric A Hetland, Mark Simons, Mark Turner, Ya-Ju Hsu, Emanuele Casarotti

Seismo Lab, Caltech

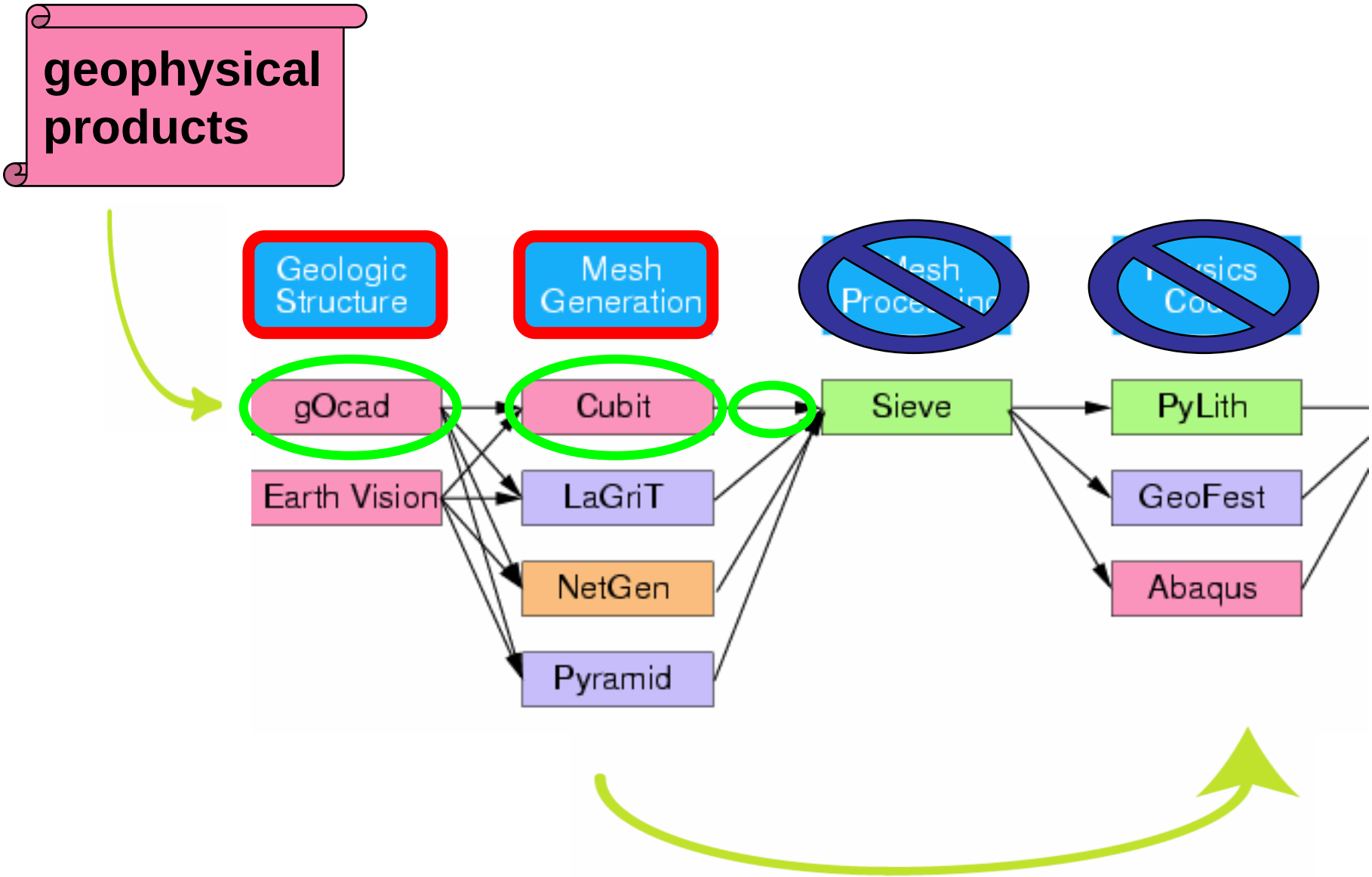
CFEM workflow:



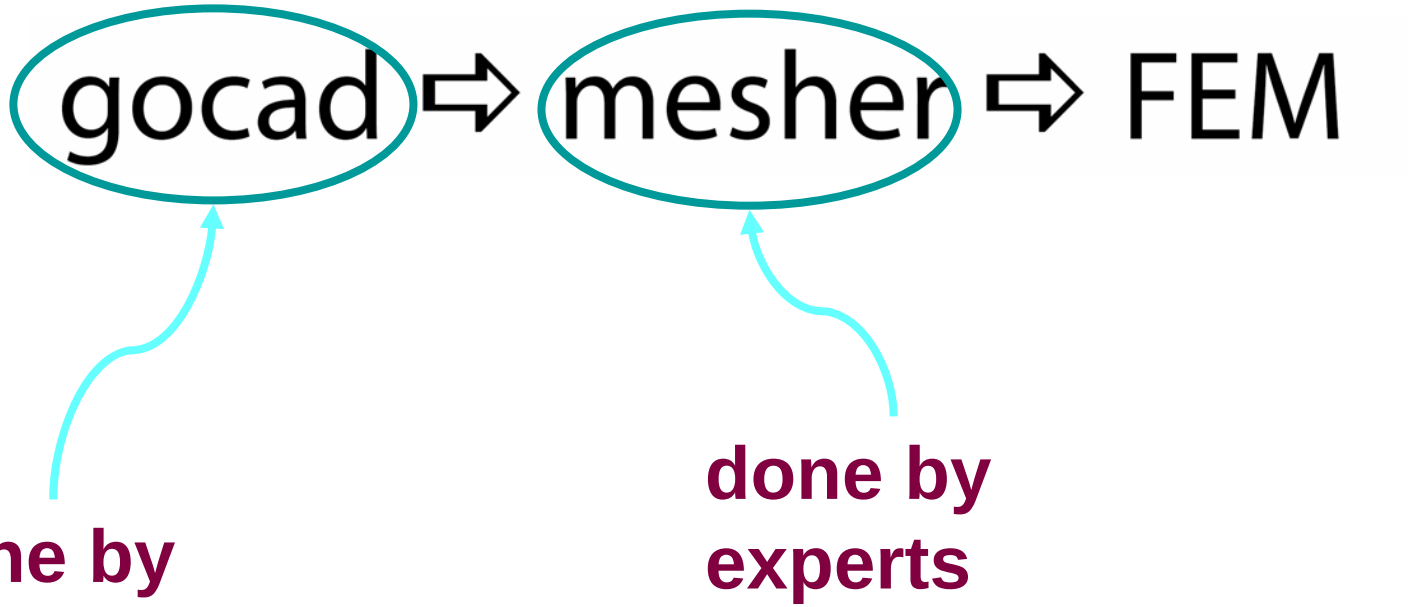
not all are equal

and these boxes do not always “talk” nicely to each other

a small piece of the workflow:



simplified workflow:

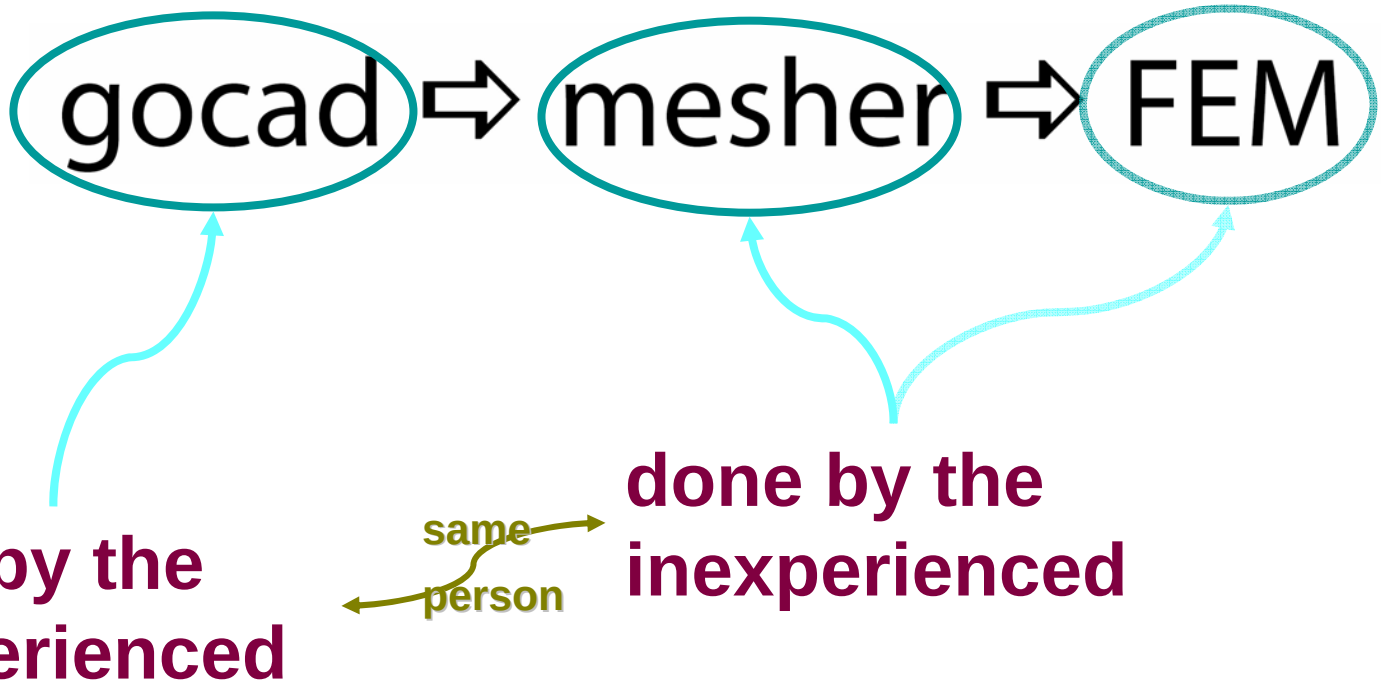


meshing expert takes output from the gocad expert, some back-and-forth…

communication: each person not entirely in fully aware of work in the other step, there may be replicated tasks…

costly: need to be able to pay the experts to do the work…

simplified workflow (the way we are doing it):



in theory, one person can do the entire work-flow

with help from experts and the experienced

inexperience = not necessarily the best way to do things

motivations, etc:

wanted to apply this process to non-SoCal regions

wanted to communicate with collaborators not using gocad (ie. translations to/from GMT, matlab, …)

key starting considerations:

define consistent projection systems (use gmt) accessible throughout the workflow, with ability to re-project at any step

use hex based, unstructured, meshes (our biases due to numerical issues)

minimize replicating tasks in gocad and cubit

define conversion steps so as to preserve as much information as possible from gocad to cubit

script-based tools/utilities define the conversion process (portable, open)

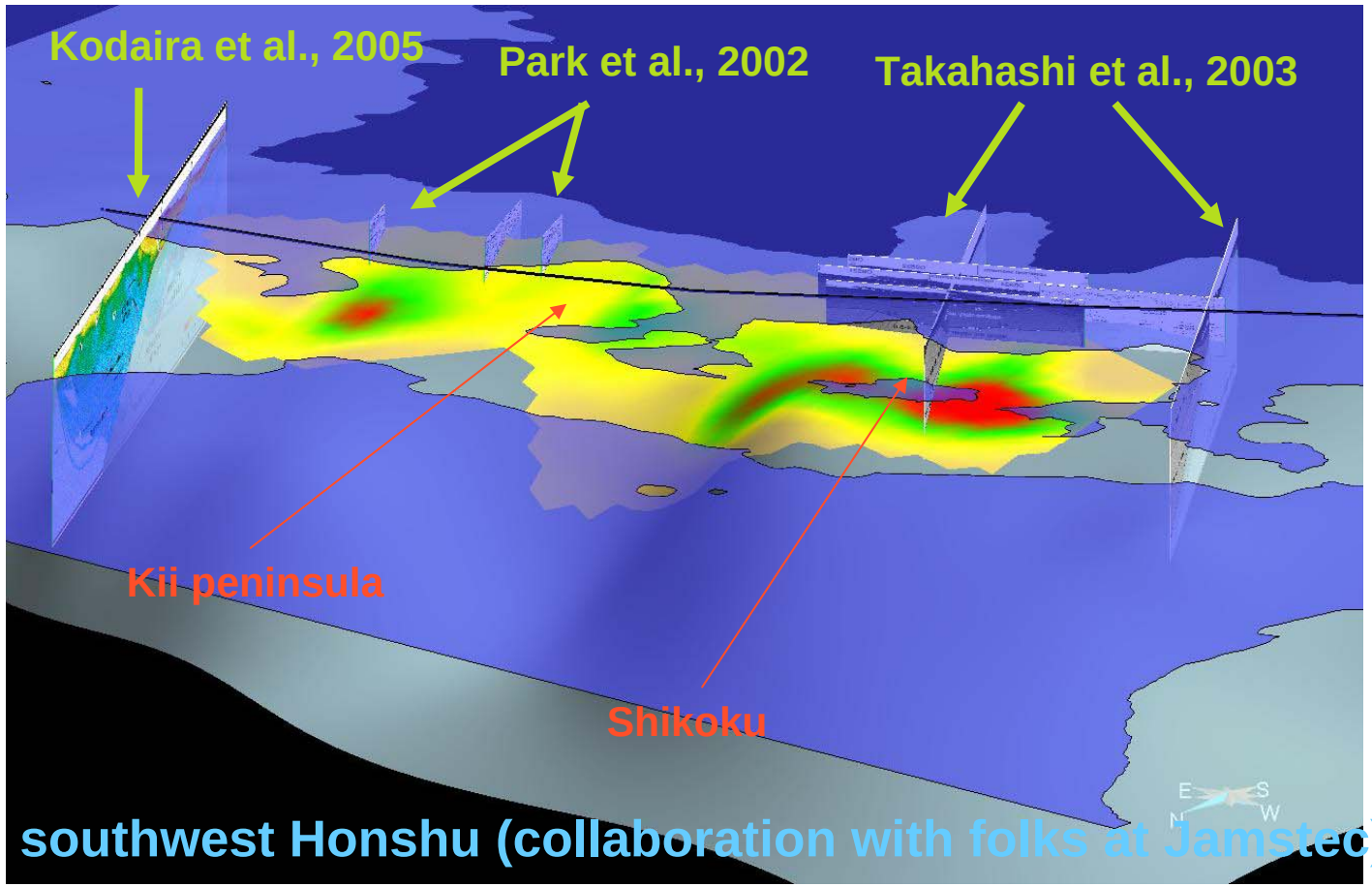
integrate (the final steps) into CIG

all comes down to book-keeping

? ⇨ gocad ⇨ cubit ⇨ pylith

motivating application – Japan (Japan & Nankai trenches):

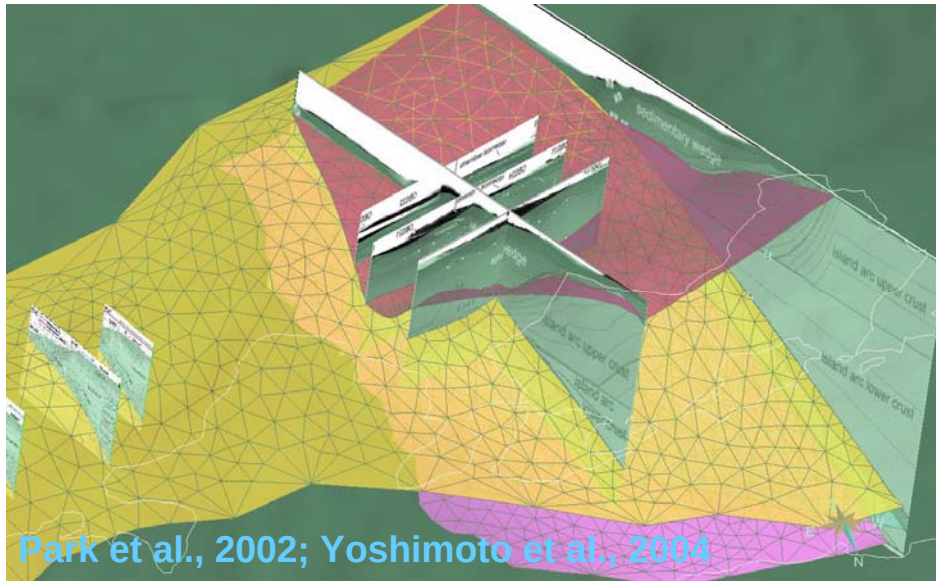
structural model: surfaces fit to constraints (primarily geophysical interpretations, wide-angle profiles, OBS, etc).



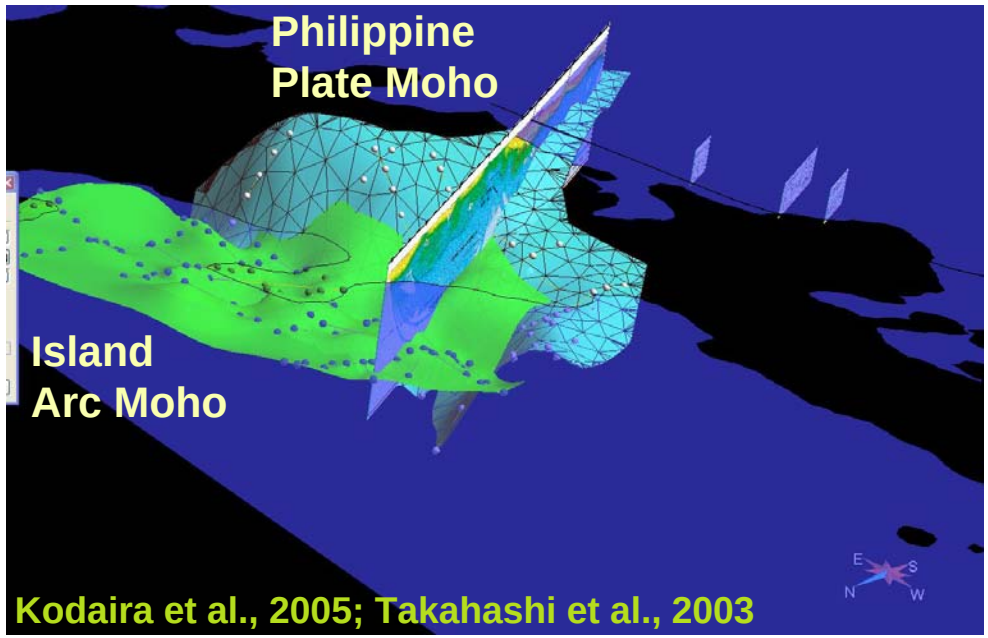
1944 Tonankai & 1946 Nankai slip models from Baba et al. (2002), Baba and Cummins (2005), slab model modified from same sources

examples of lithospheric structure constraints:

active source seismology:
wide-angle reflection/refraction lines (onshore&offshore), MCS lines (non-commercial products)

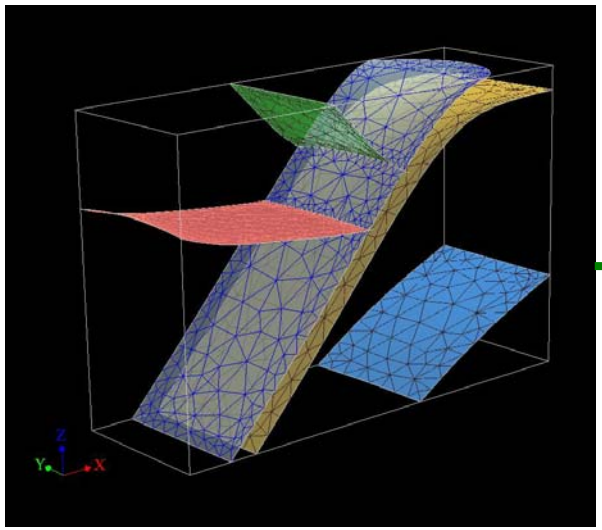


passive source seismology:
receiver functions, etc…



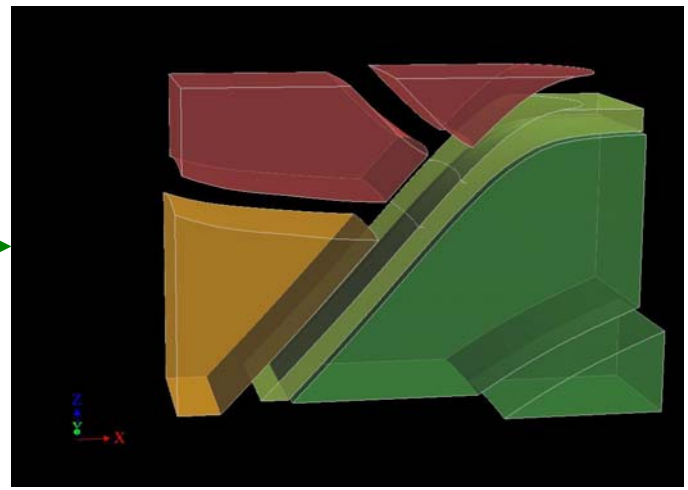
structural model vs. solid model:

structural model (free surfaces)

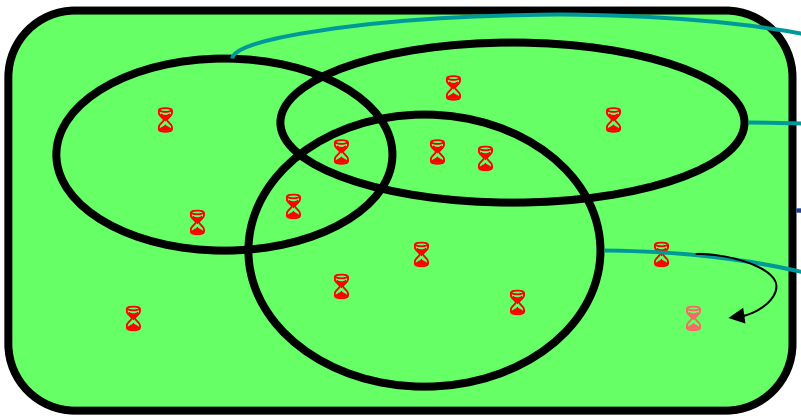


well-defined &
fairly automated

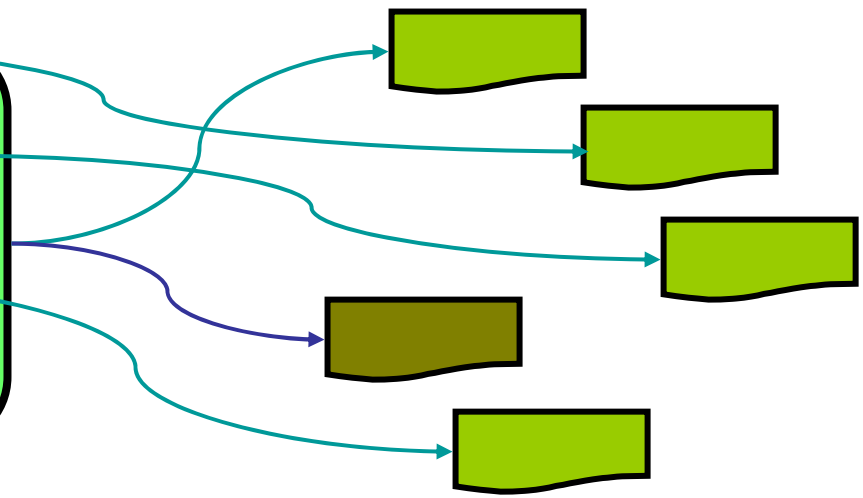
solid model (closed volumes)



structural model



solid models



? ⇨ gocad: ? = geological and geophysical data/results

set of tools (perl scripts, with some matlab, gmt, etc, as needed)

- keeps track of (lat,lon,z) → (x,y,z) mapping for each project
- translate “legacy” file formats into gocad objects, and back again
- transfer to/from gocad objects for collaboration

mostly done or well defined

done, but will “integrate” with A. Plesch’s beach-ball wizard

just started

solved in gocad
(need to project image coords and locate it “manually”)

external information	examples	file types	gocad object
geologic	topography discontinuities faults	GMT (*.grd) xyz files	TSurfs
geophysical	receiver funcs. MCS profiles w/a profiles	v.in files ascii ???	Curves/TSurfs Voxets/SGrids
catalogs	seismicity GPS locations	multi-column	PointSets
2D props	slip models gravity profiles	GMT (*.grd) xyz files multi-column	2DGrids
3D props	tomography density Q, Vp/Vs	xyz files ???	SGrids
images	geophysical interpretations	jpegs	Voxets

gocad ⇨ cubit callenges:

topology book-keeping in gocad is very different from book-keeping in most FE meshers

→ each TSurf in gocad "owns" its own vertexes and borders, adjoining TSurfs do not reference common nodes

triangular faceted surfaces are read into cubit as FE meshes, cubit treats the facets as finite elements and extracts geometry from mesh

→ resulting surfaces are in the MESH based geometry engine, ACIS engine much better

→ surface normals are not preserved!

→ very little control over extraction of geometry in cubit

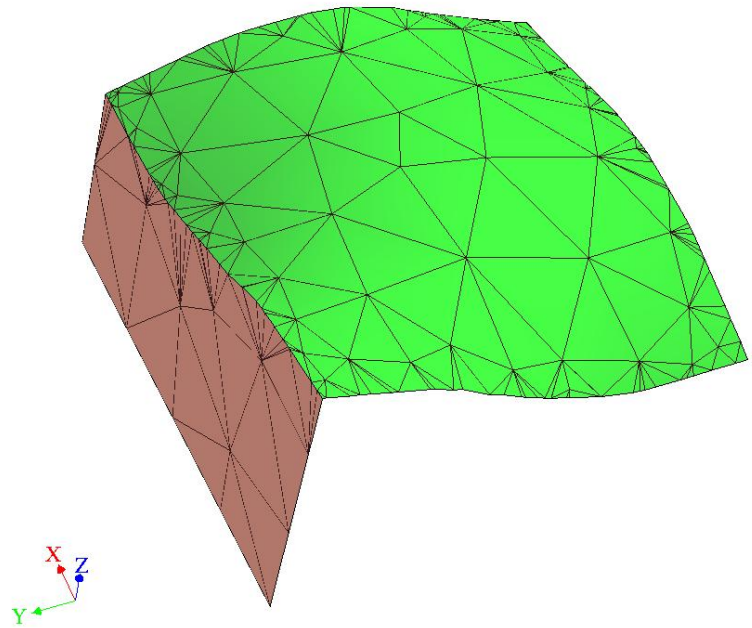
ACIS format is proprietary

→ can not directly write gocad objects into ACIS formats

→ triangle facetted surfaces do not exist in ACIS engine

touching gocad surfaces:

these two TSurfs touch, but each surface references different, collocated vertexes

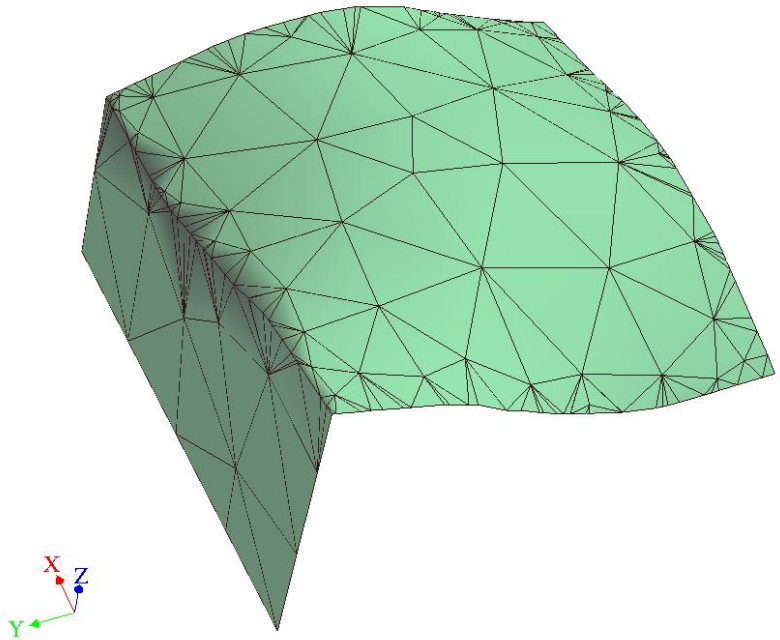


gocad commands:

→ create new TSurf from 2 TSurfs

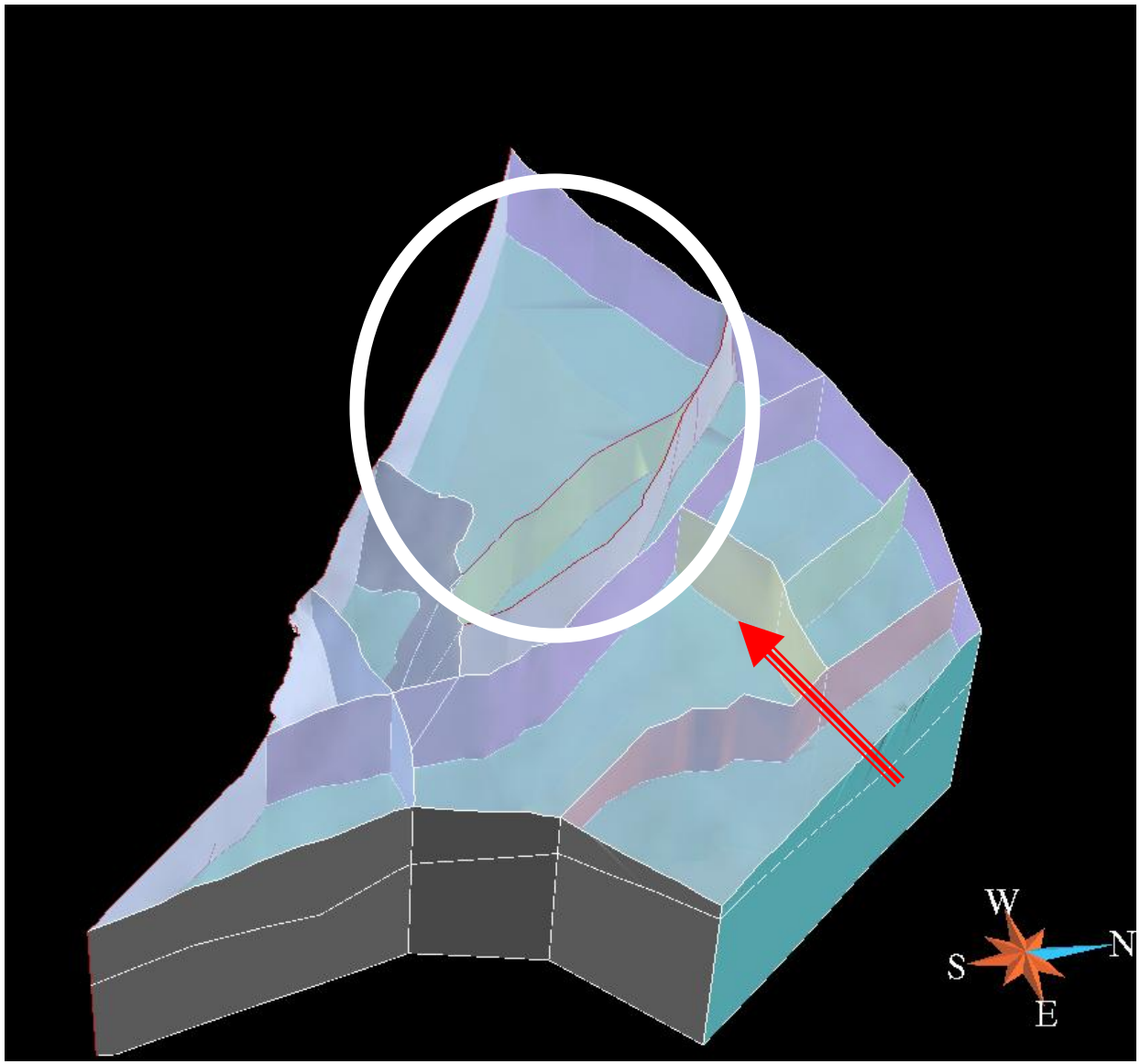
→ merge TSurf parts

now it is a single TSurf, with common nodes along the joint

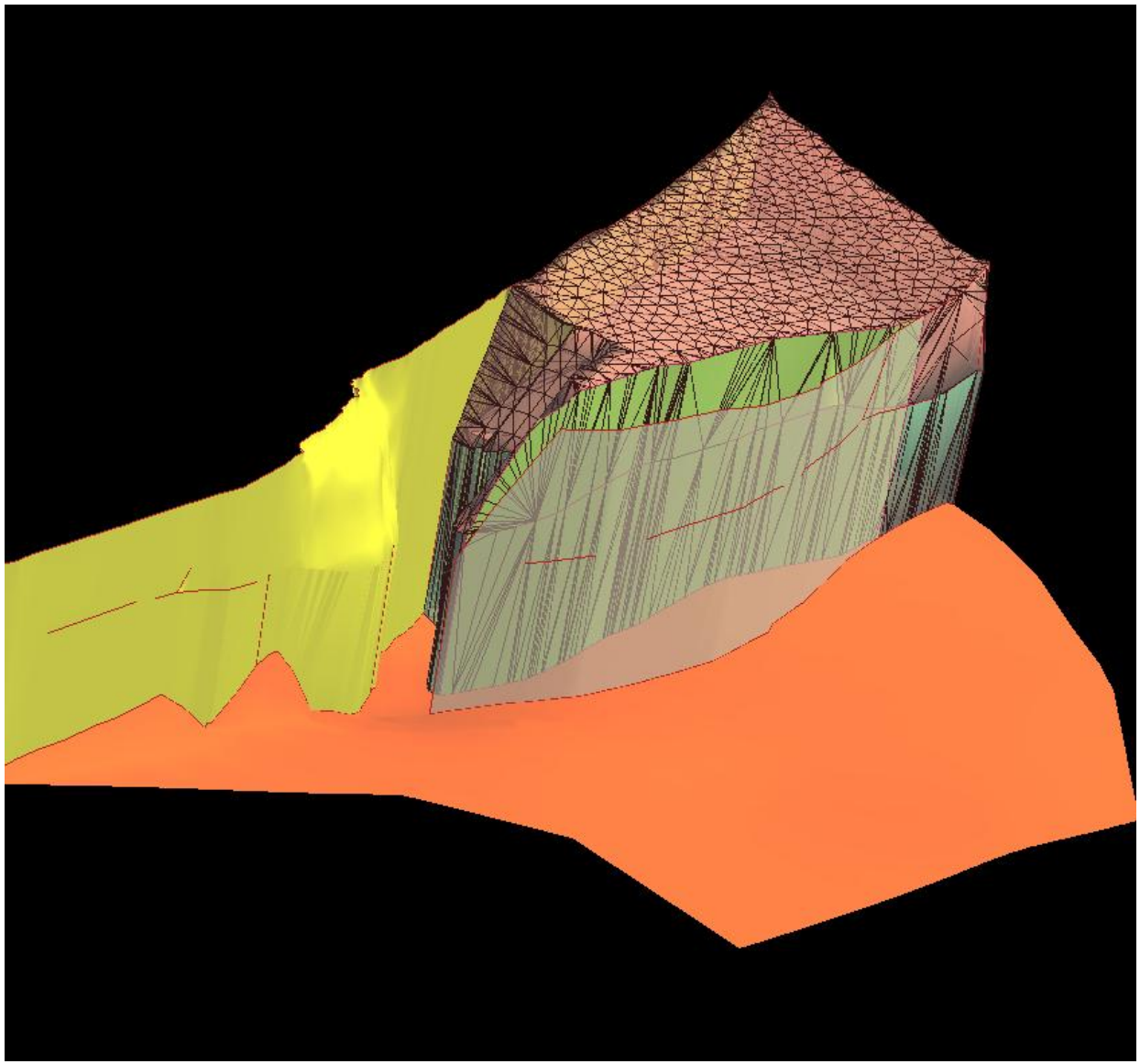


the CBM format (as I understood in 06/15/2006):

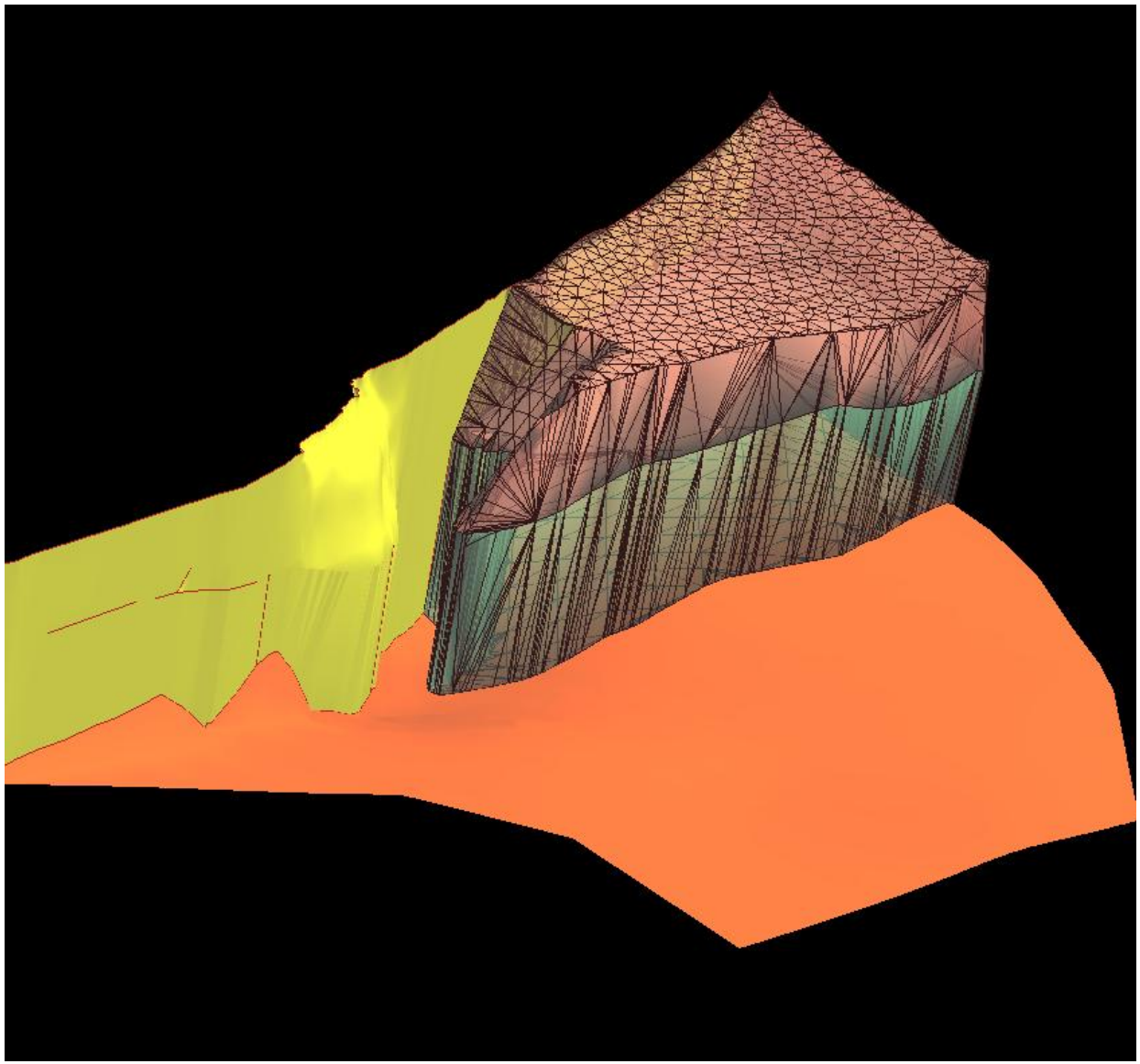
surface parts that define a block are joined into one, *closed* surface…



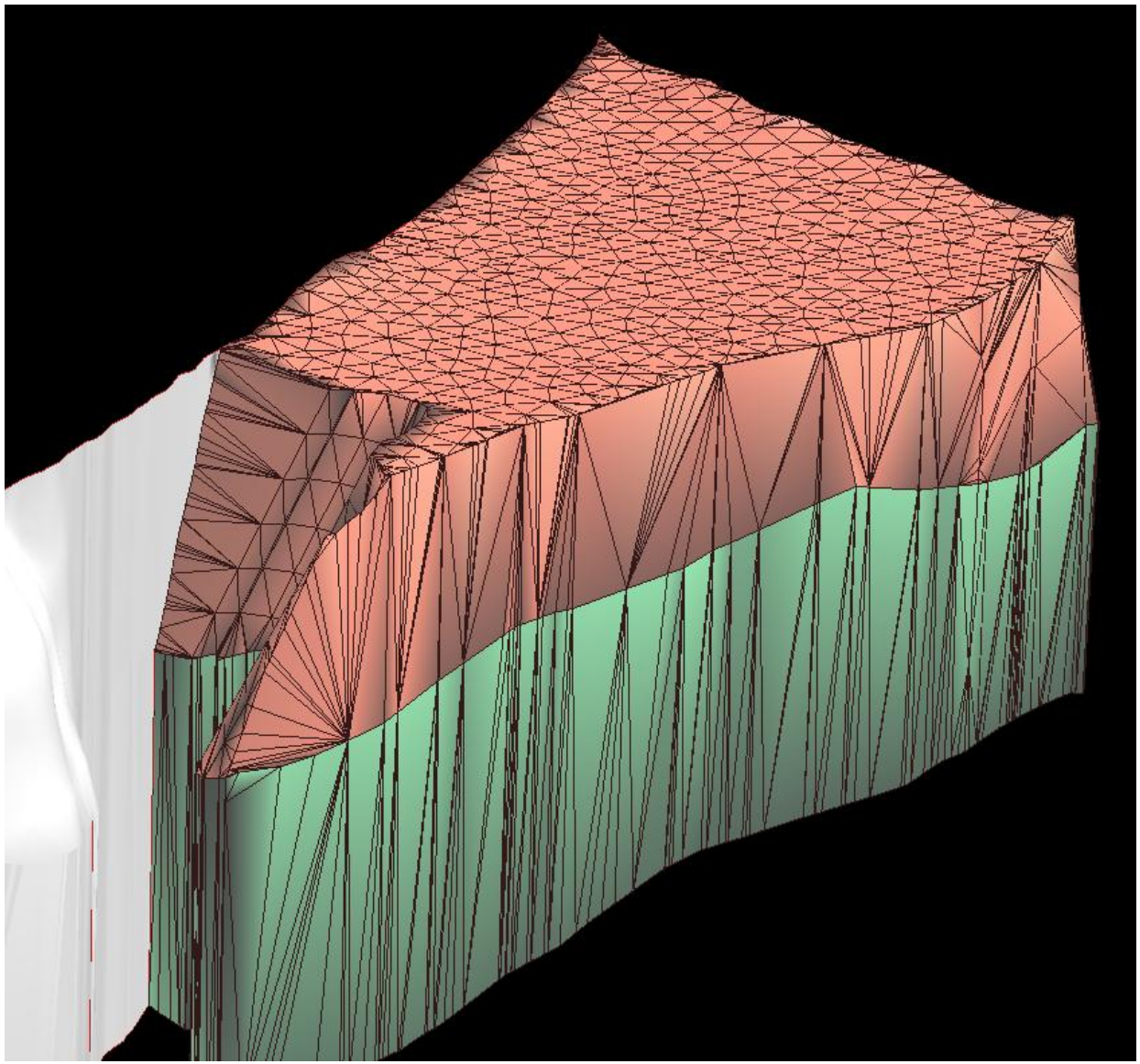
the CBM format (as I understood in 06/15/2006):



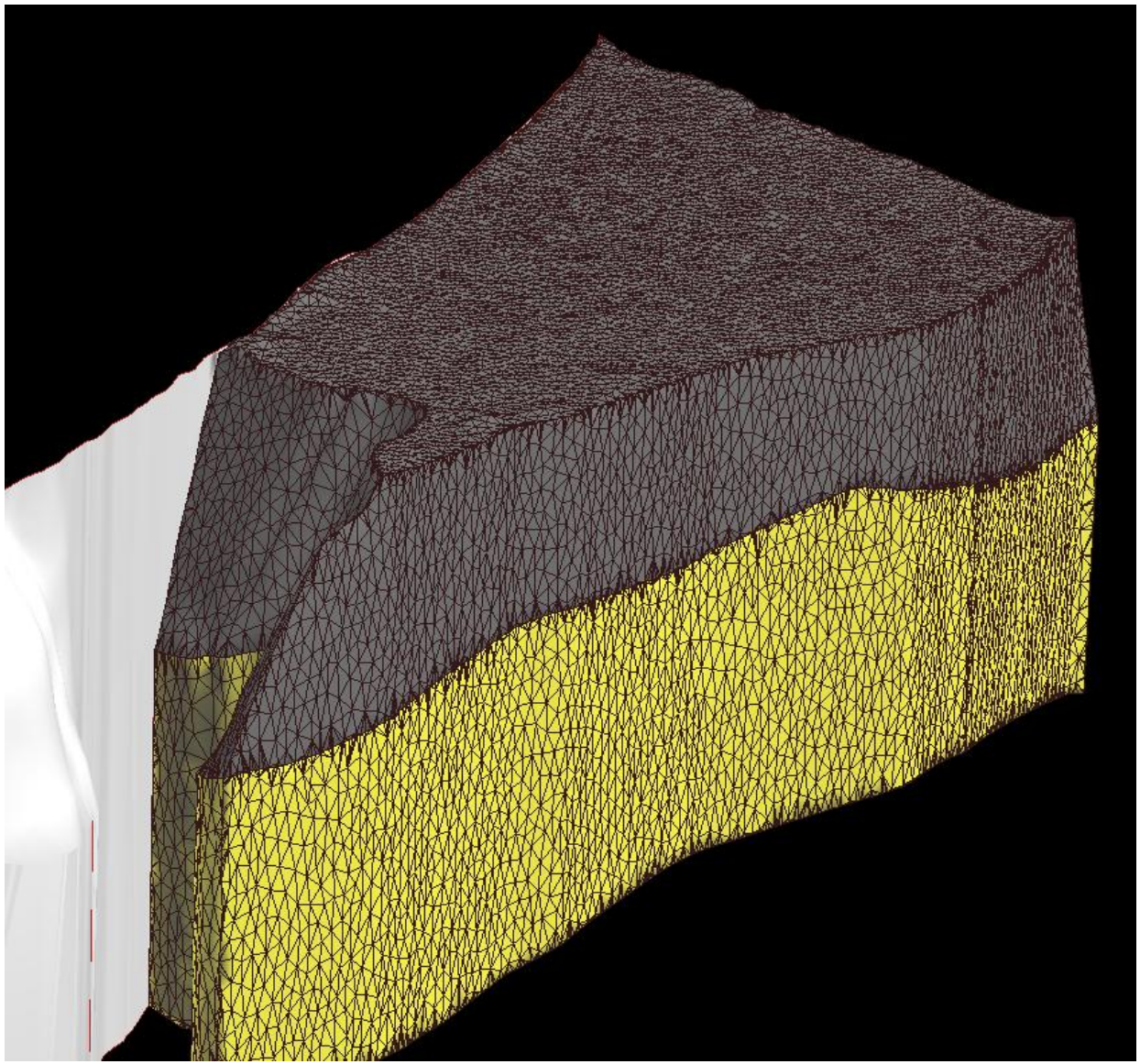
the CBM format (as I understood in 06/15/2006):



the CBM format (as I understood in 06/15/2006):



the CBM format (as I understood in 06/15/2006):



the CBM format (as I understood in 06/15/2006):

each vo sed (and if
parts w ged, each side
of the v he shares vertexes
along the edges), but
adjoining volumes do not
share a common surface,
border surface is duplicated…

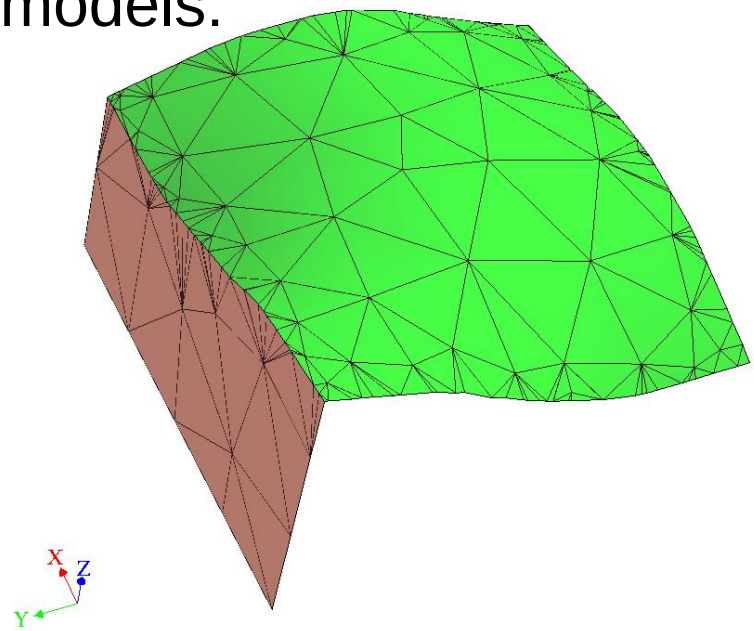
our approach to transferring structural models:

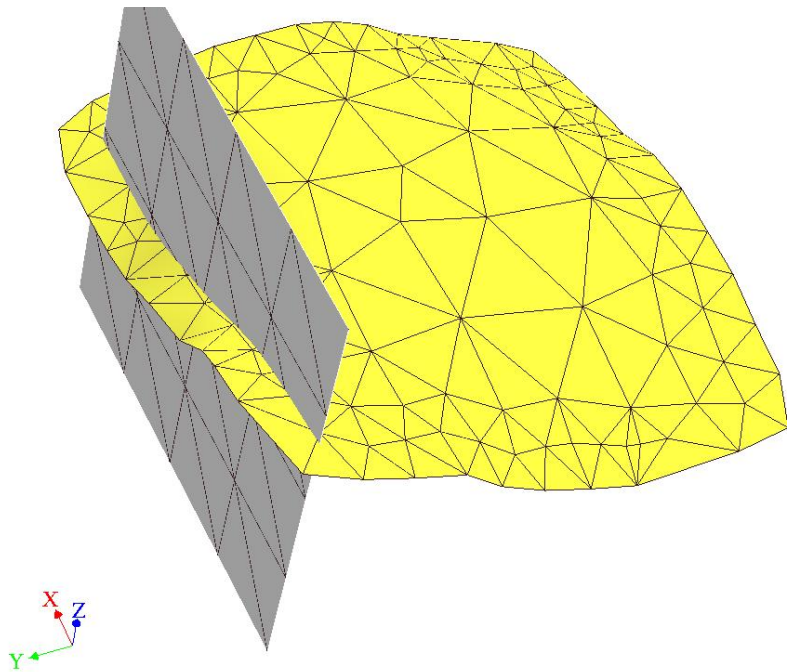
We do not constrain our workflow to deal with only the CBM format – *our goal is not to replicate the CFM/CBM for other regions*…

We create our **structural model** so that surfaces that might intersect are overlapping (easy to extend surfaces in gocad), and do not duplicate surfaces…

We "assemble" the **solid model** (i.e., closed volumes) in gocad (using Model3d objects) or in cubit, or both…

We try to keep creation of the solid model simple, so we can re-create solid models as we tweak the interpreted lithospheric structure – *we do not have much faith in our initial structural models, and* ***avoid having to define a comprehensive structural model before ⇨ cubit****…*





gocad ⇨ cubit fundamentals:

- TSurfs/TFaces are read into cubit as faceted surfaces (mesh based geometries)
- faceted surfaces are meshed using a mapped meshing scheme
- create new ACIS-geometry surfaces from the meshed faceted surfaces
- assemble the ACIS surfaces to the level of a given gocad model (from just free surfaces, to closed, joined volumes)
- process is “automated” in a perl script – requires some error checking, and problems are known to exist
- almost all problems are fixed using existing TSurf cleaning functions in gocad, other problems fixed by changing run-time options in the conversion script
- script actually writes a series of cubit commands to a *.jou file

icon for “run script + play cubit journal file”: 

TSurf ⇨ mapped mesh-based surfaces:

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

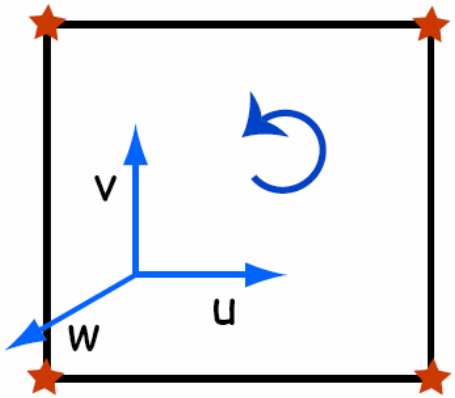
seem to have very little direct control over the key parts of this process

surfaces are read into cubit as triangle facetted surfaces, then overlaid with regular nets (mapped scheme mesh), the intersections of the nets become knots in the new surfaces in the ACIS-geometry engine…

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

I think… (remember, I am not the expert)

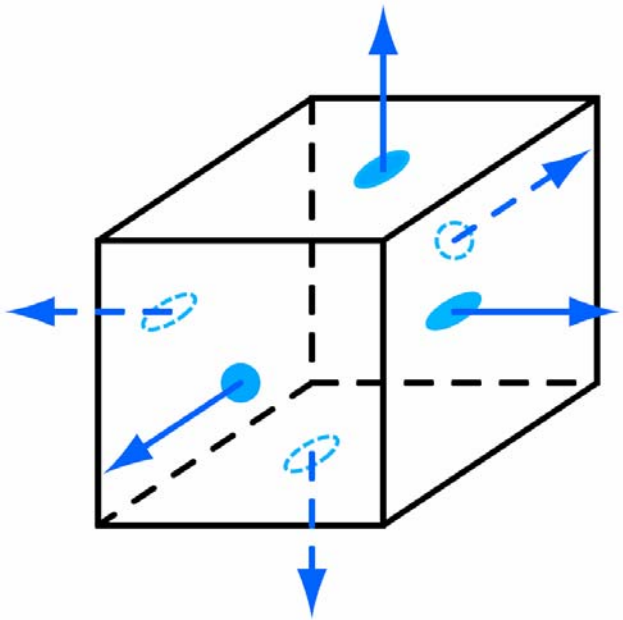
aside on orientation:



in a FE mesher all surfaces have corners and orientations

volumes also need to have orientations to mesh nicely

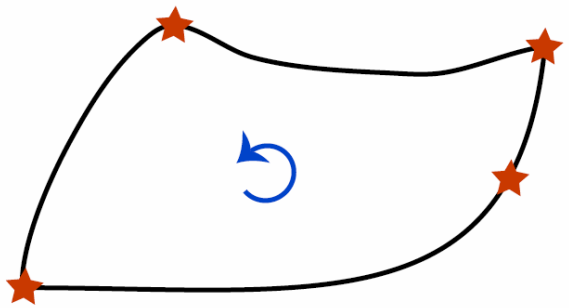
(in this case all surface normals point out)



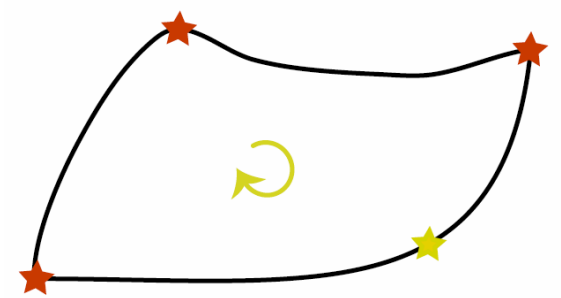
gocad also orients surfaces and volumes in a Model3d object, as well as defines corners on surfaces (the corners can always be redefined, if needed)

this information is lost when surfaces are imported into cubit as facetted surfaces

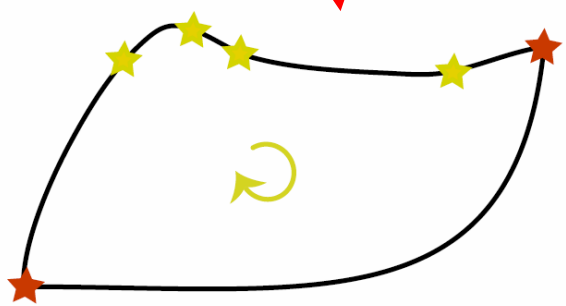
an example:



surface with 4 corners
and orientation



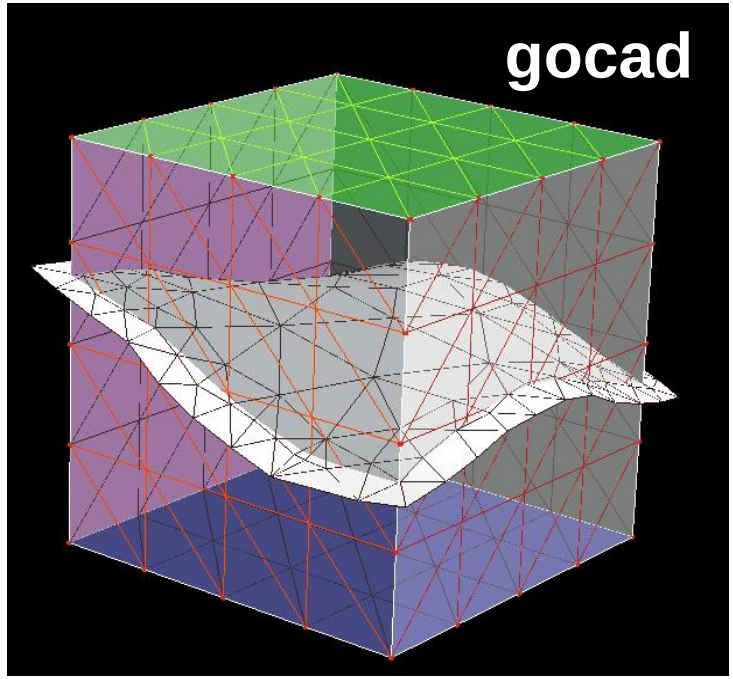
not so bad, only
misplaced one corner,
and lost the orientation…



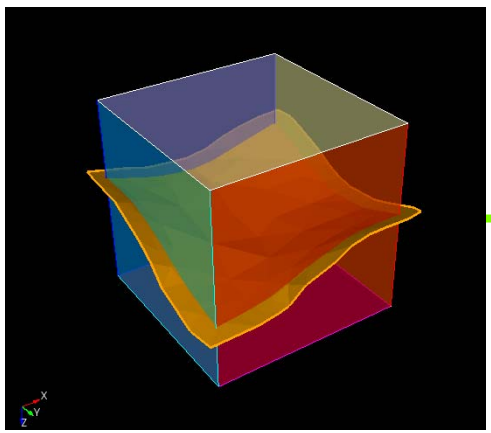
quite bad, especially the
missed corner, this will be
hard to convert to ACIS
object in cubit…

the loss of the surface orientation is probably more of a problem…

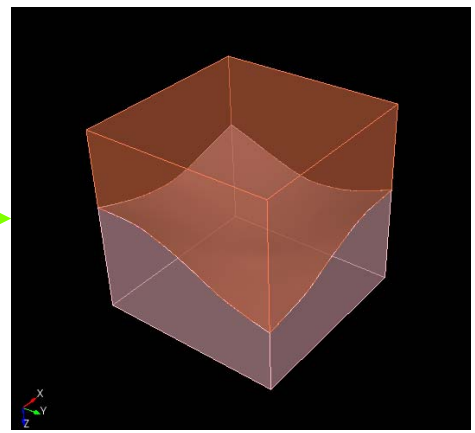
1 - create volumes in cubit:



in gocad each TSurf is treated as a separate geometric entity that will be translated to a separate surface in cubit (in the ACIS engine)



cubit commands

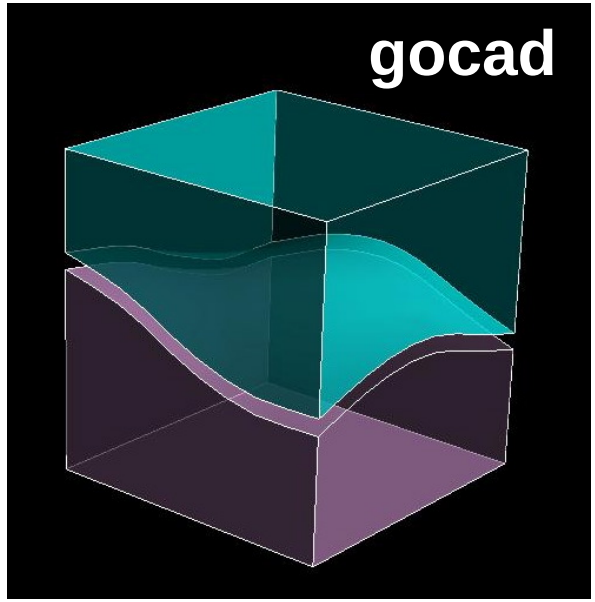


7 surfaces in cubit

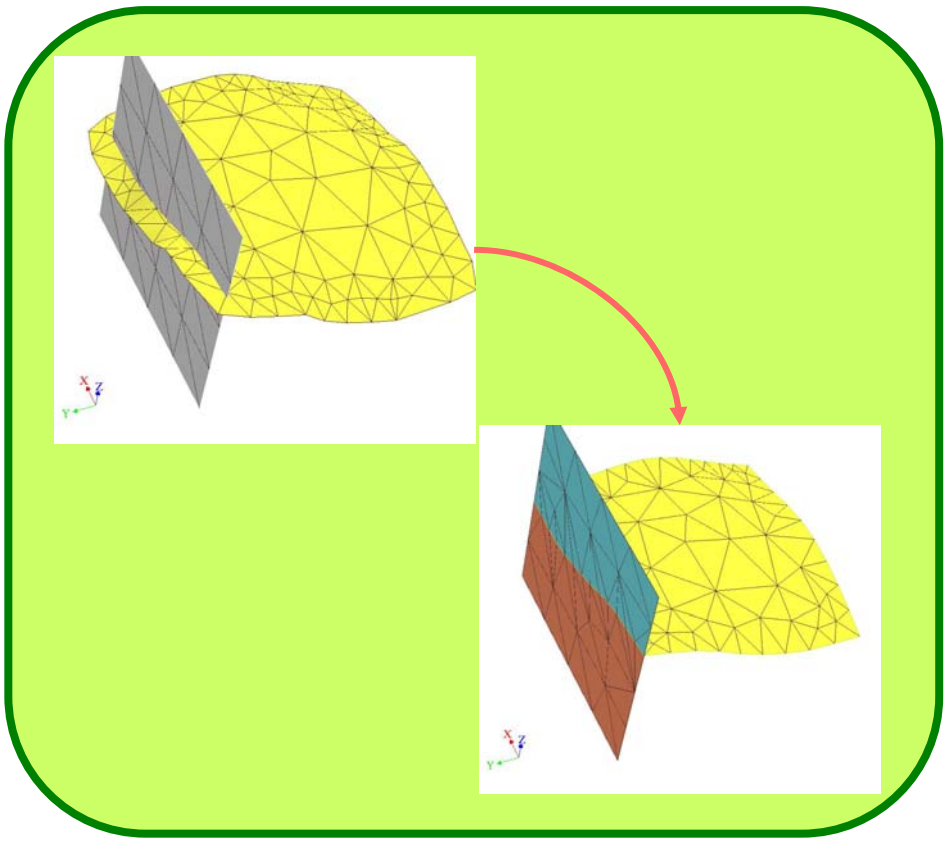
2 volumes in cubit

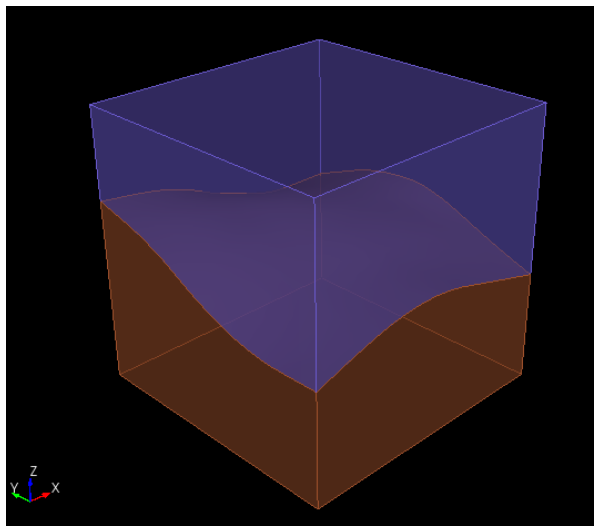
2 - create volumes in gocad:

two closed volumes in gocad



create Model3d object in gocad from (overlapping) TSurfs

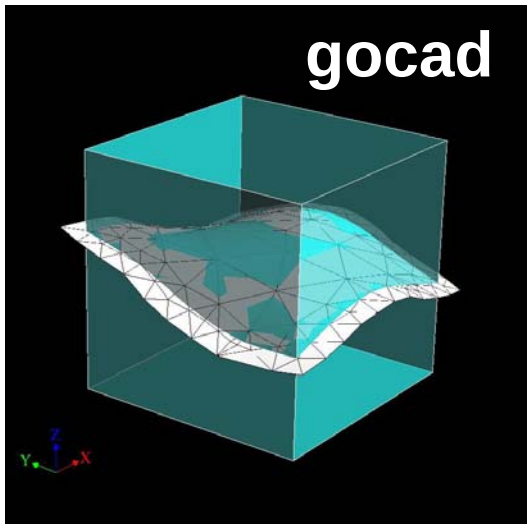




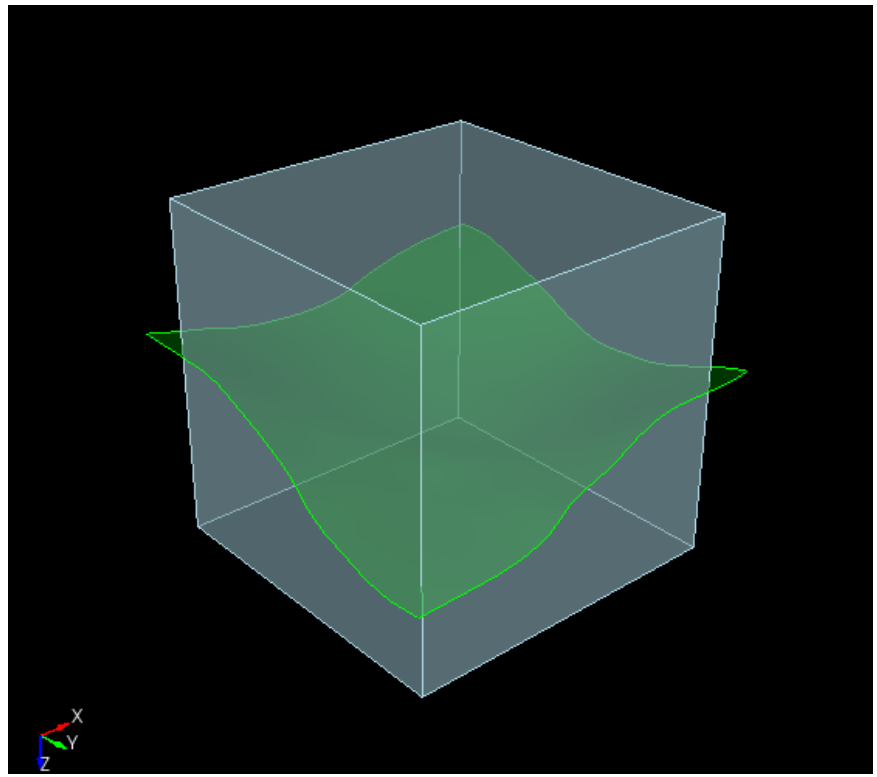
two closed volumes in cubit

3 - hybrid:

create one volume in a Model3d object in gocad, to get one volume in cubit, plus import the middle surface from gocad into cubit…



1 volume and 1 TSurf in gocad



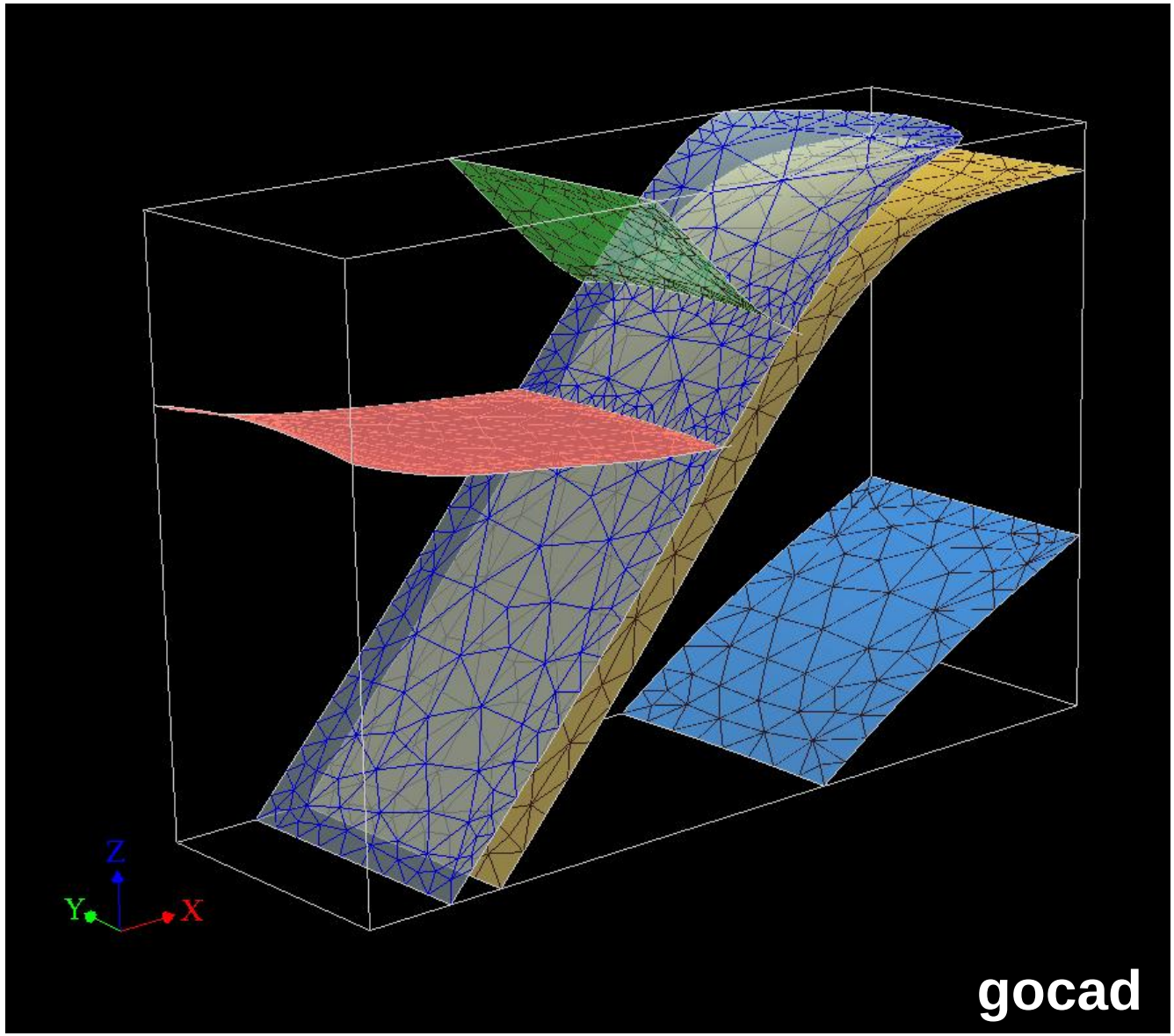
1 volume and 1 free surface in cubit

some additional cubit commands to get two volumes…

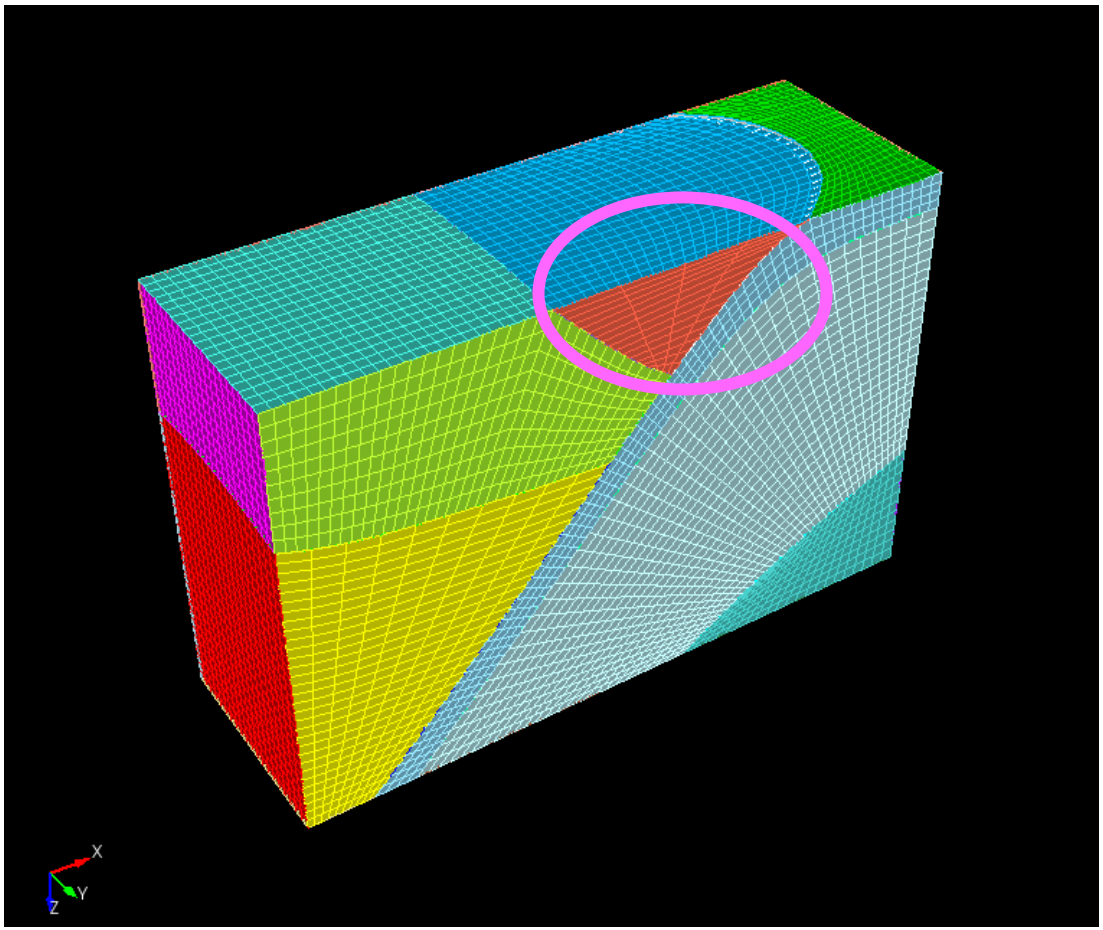
toy model in gocad:

Model3d:

- 5 internal surfaces
- 6 bounding surfaces
- 6 volumes
- 31 TSurf parts



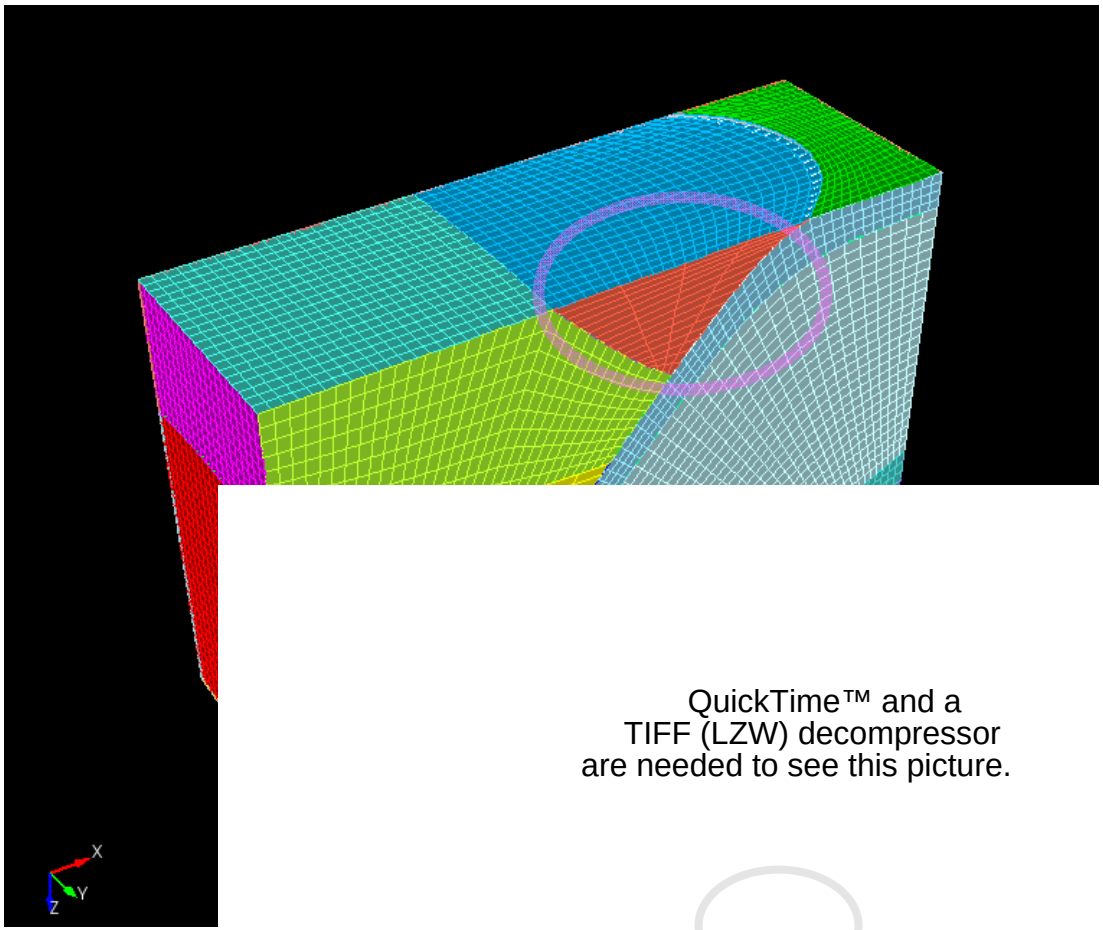
toy model in cubit:



partially

31 mapped surfaces (facetted surfaces in mesh-geometry)

toy model in cubit:



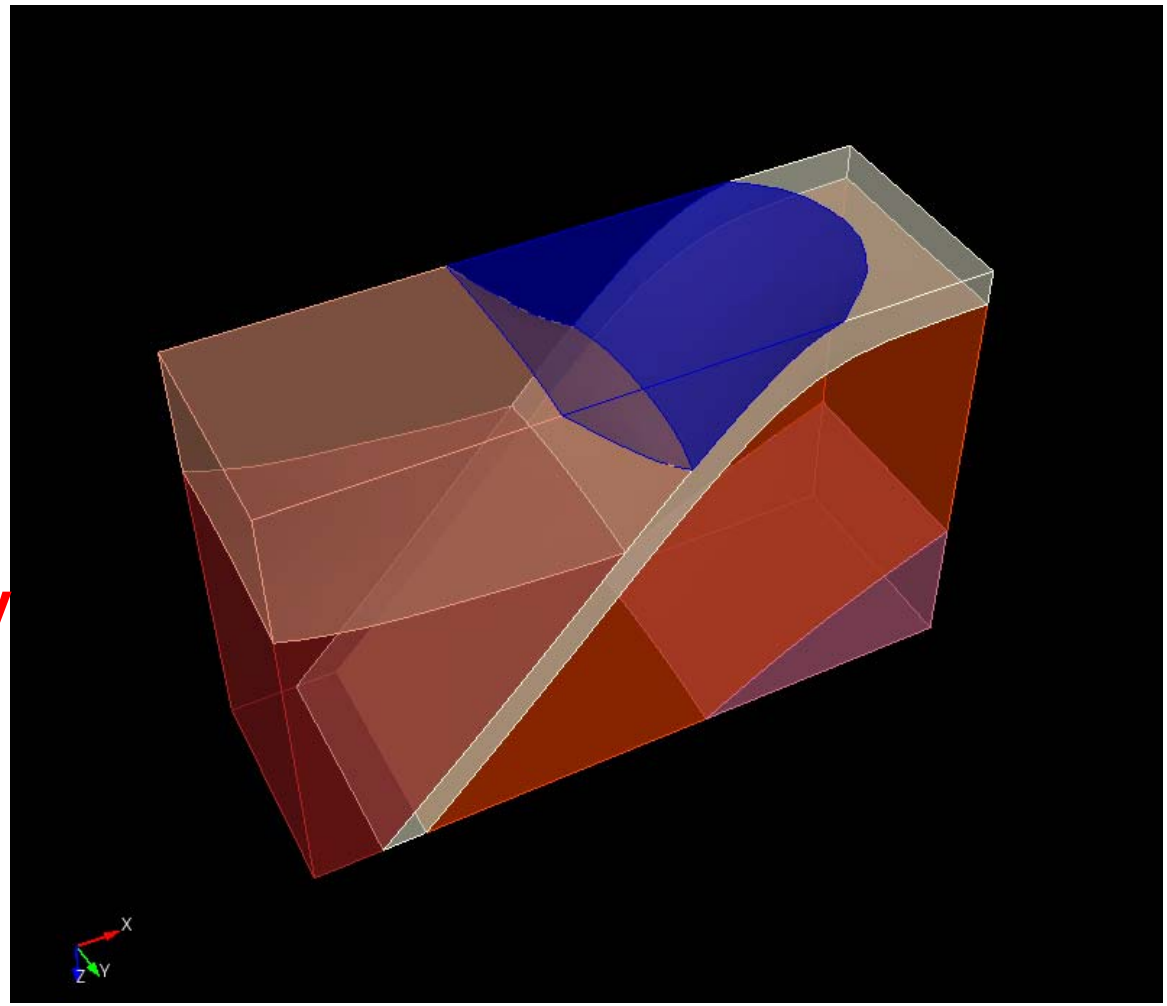
partially

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

31 mapped surfaces (facetted surfaces in mesh-geometry)

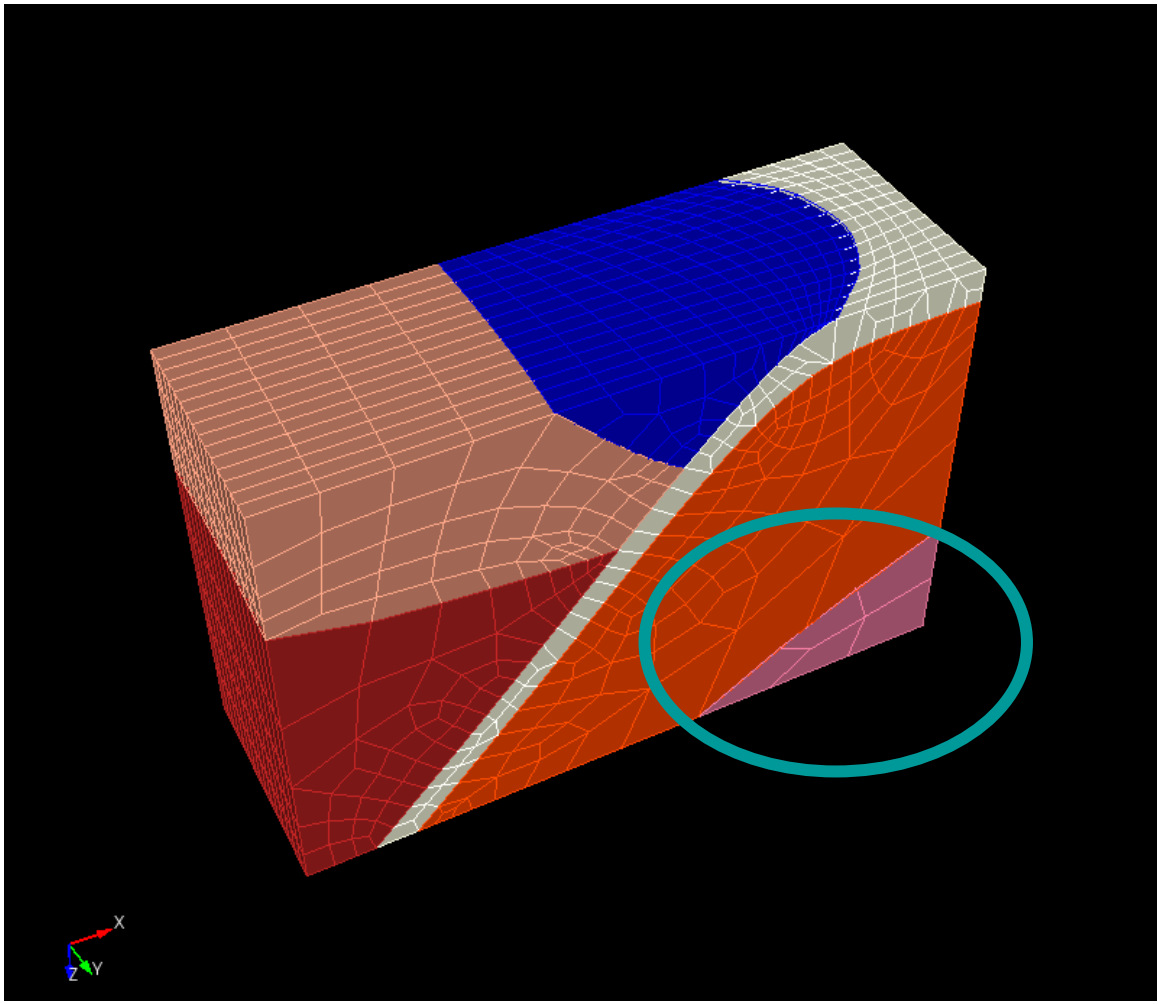
toy model in cubit:

completely



6 connected volumes (ie., sharing common surfaces, well sort of)

toy model in cubit:



cubit commands

continuous, finite element mesh (well, kind-of)

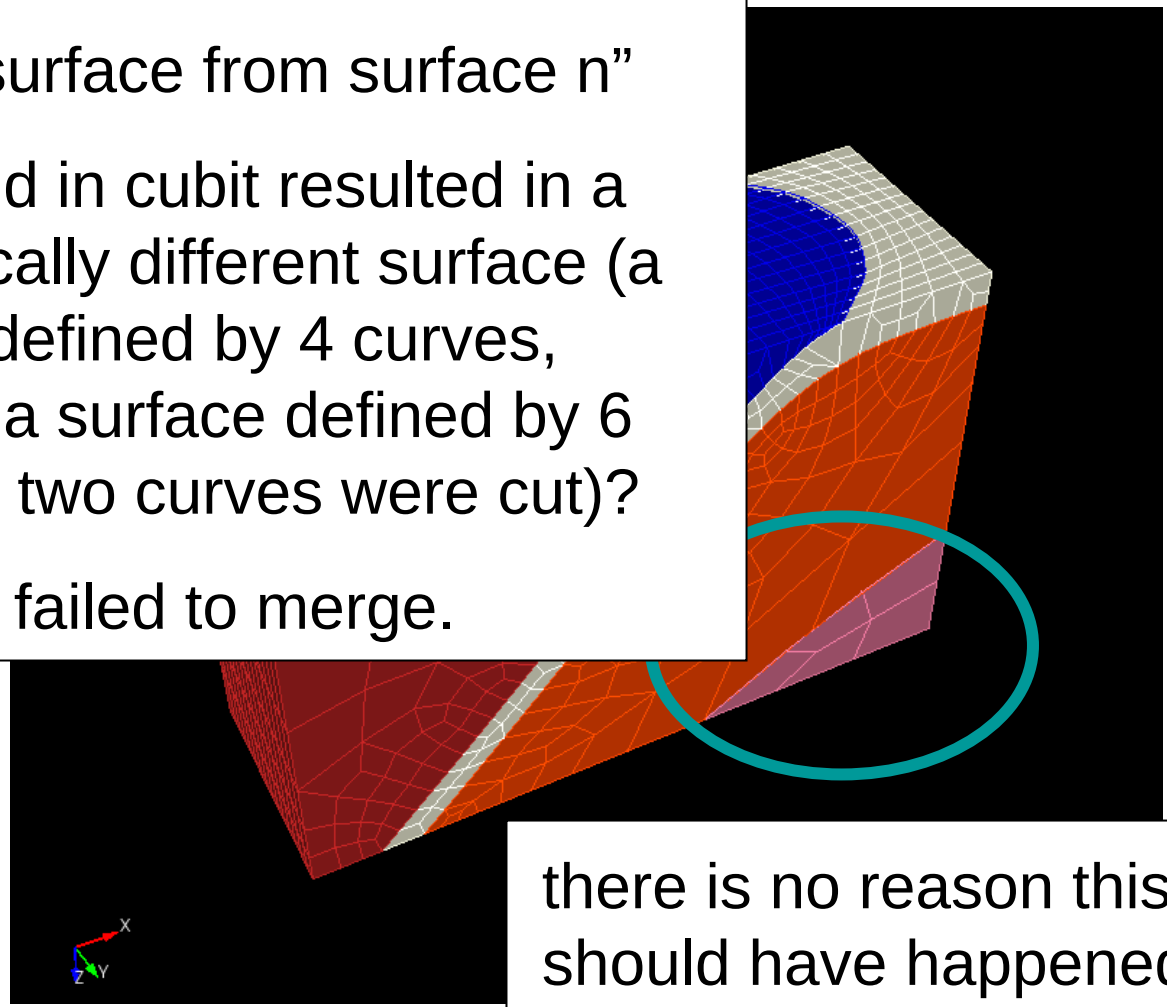
toy model in cubit:

“create surface from surface n”

command in cubit resulted in a topologically different surface (a surface defined by 4 curves, became a surface defined by 6 curves – two curves were cut)?

volumes failed to merge.

commands



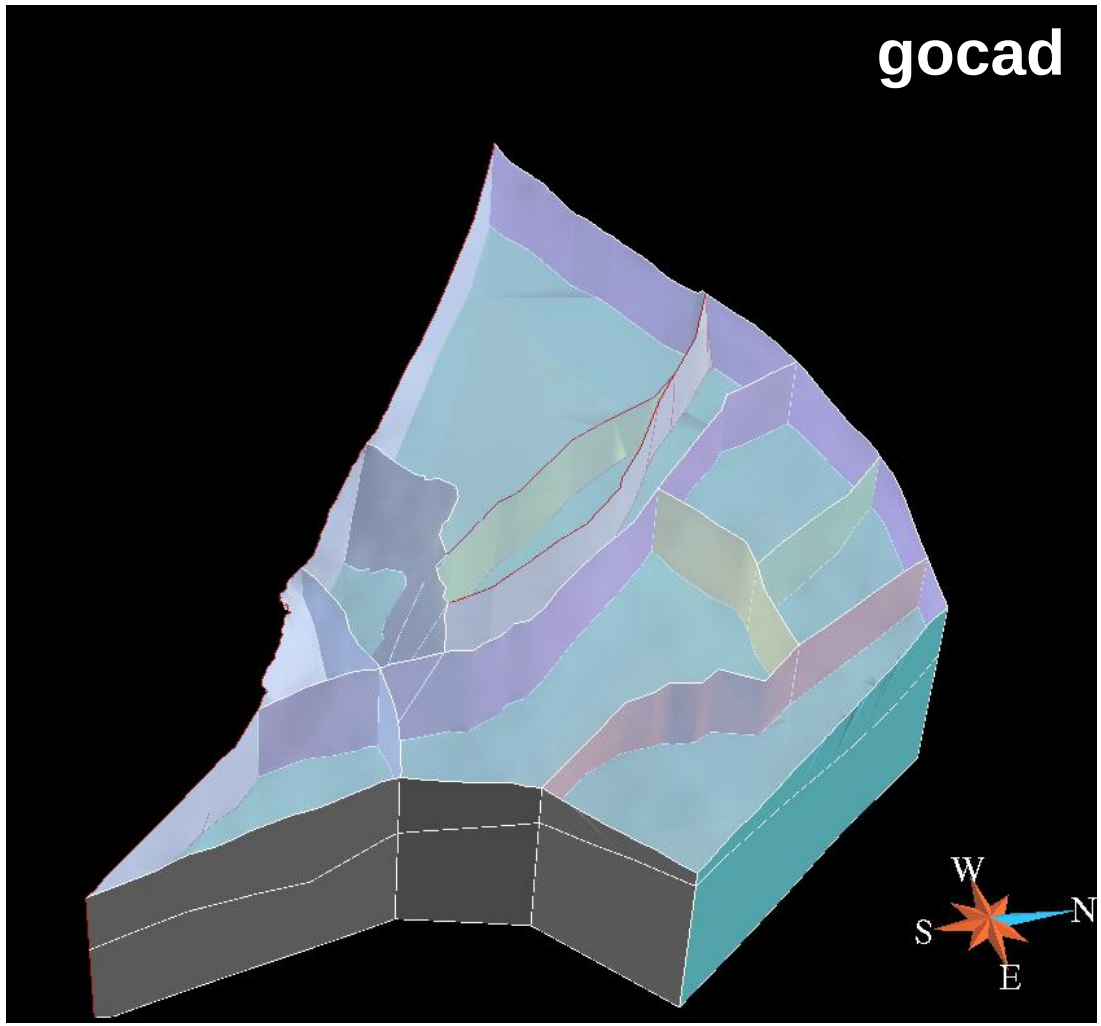
there is no reason this should have happened…

continuous, finite element mesh (well, kind-of)

back to the Mojave CBM:

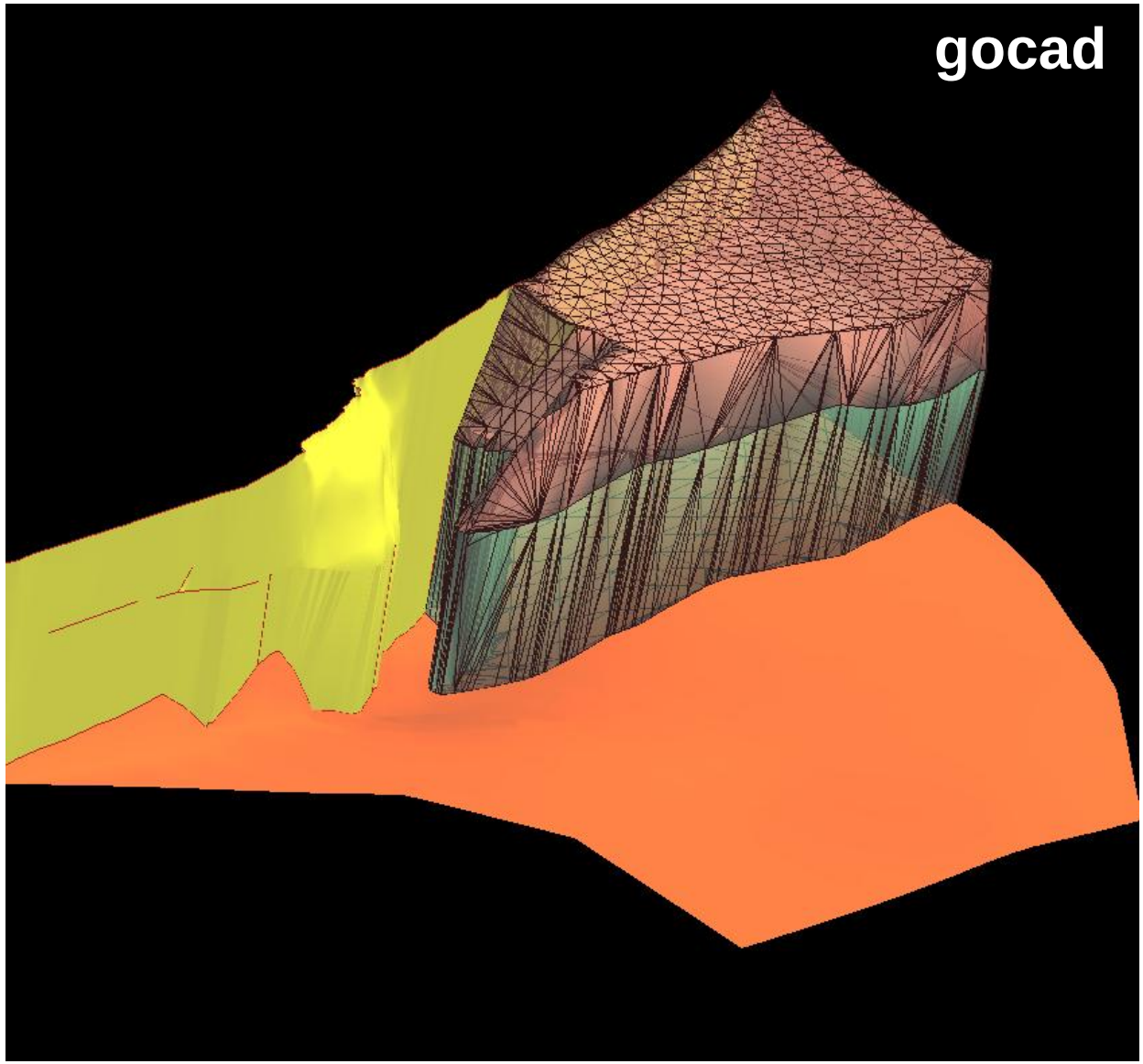
Mojave CBM is made up of closed surfaces defining blocks, but also **separate TSurfs defining the block borders** (other information as well, like the regular block grids).

To mesh the Mojave CBM in cubit, all we need to do is join separate TSurfs into larger TSurfs and extend them a bit – we need intersecting surfaces, not closed volumes…



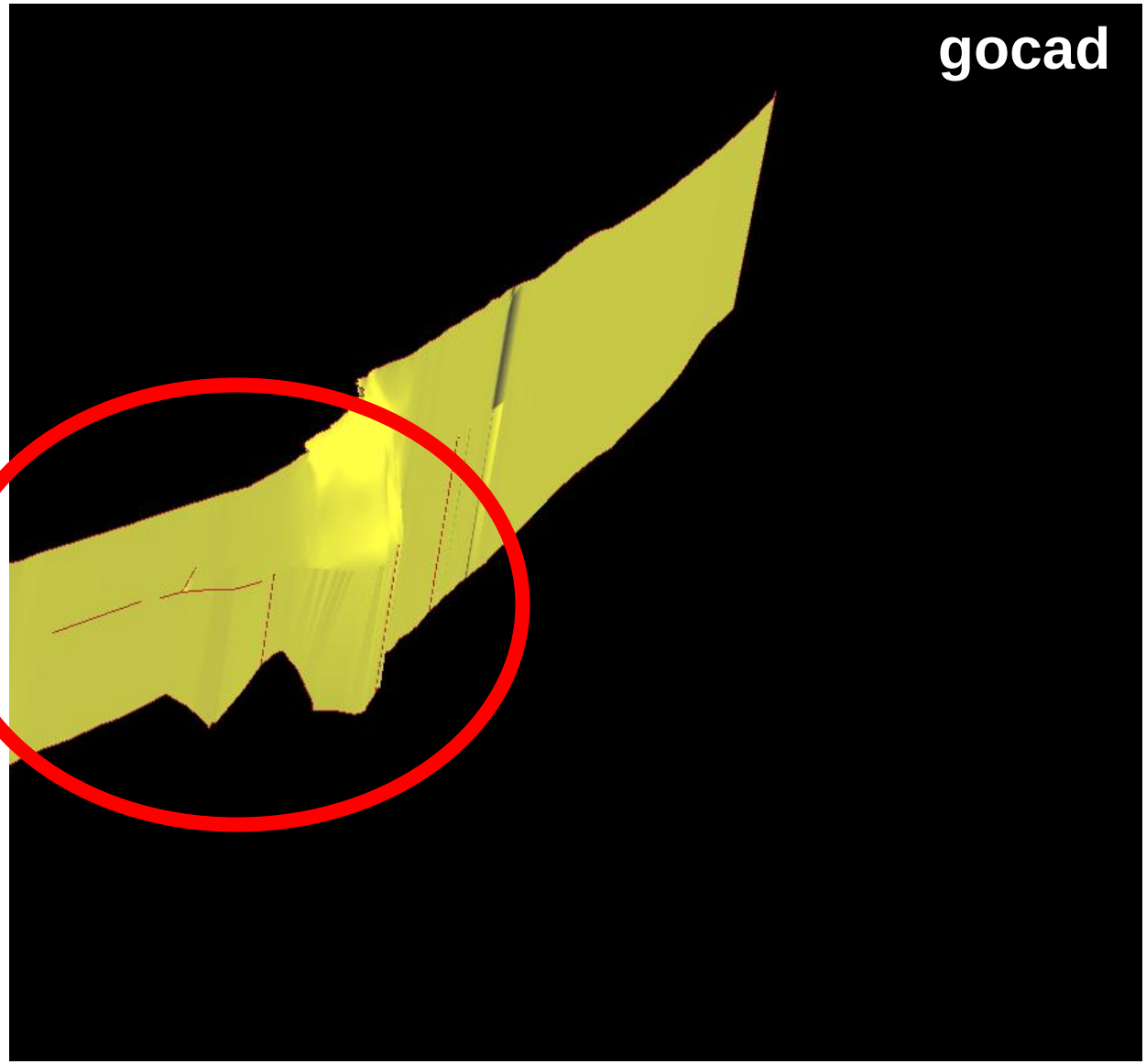
back to the Mojave CBM:

Focus on just a set of connected TSurfs, that form the western border of the model.



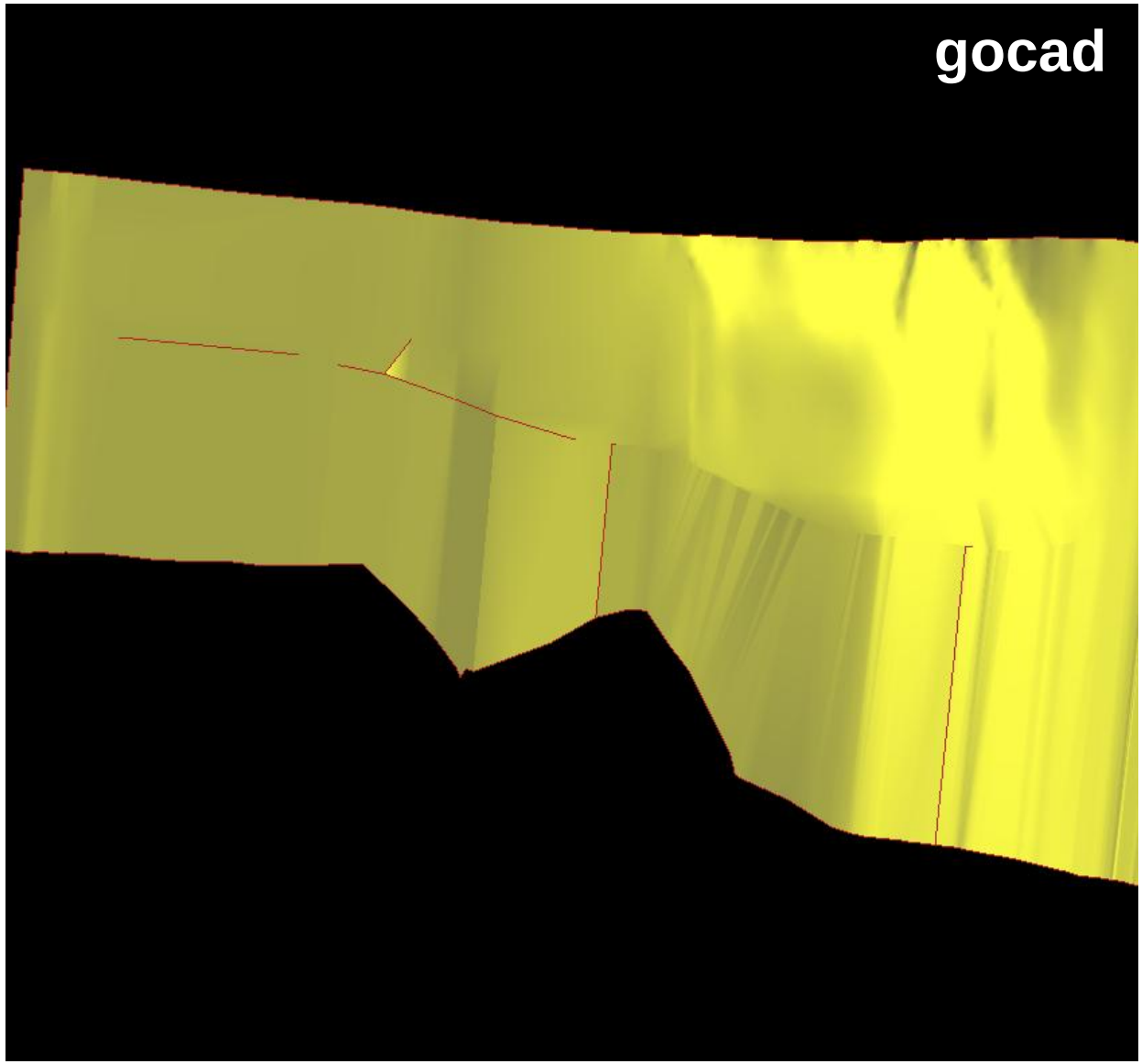
back to the Mojave CBM:

Focus on just a set of connected TSurfs, that form the western border of the model.



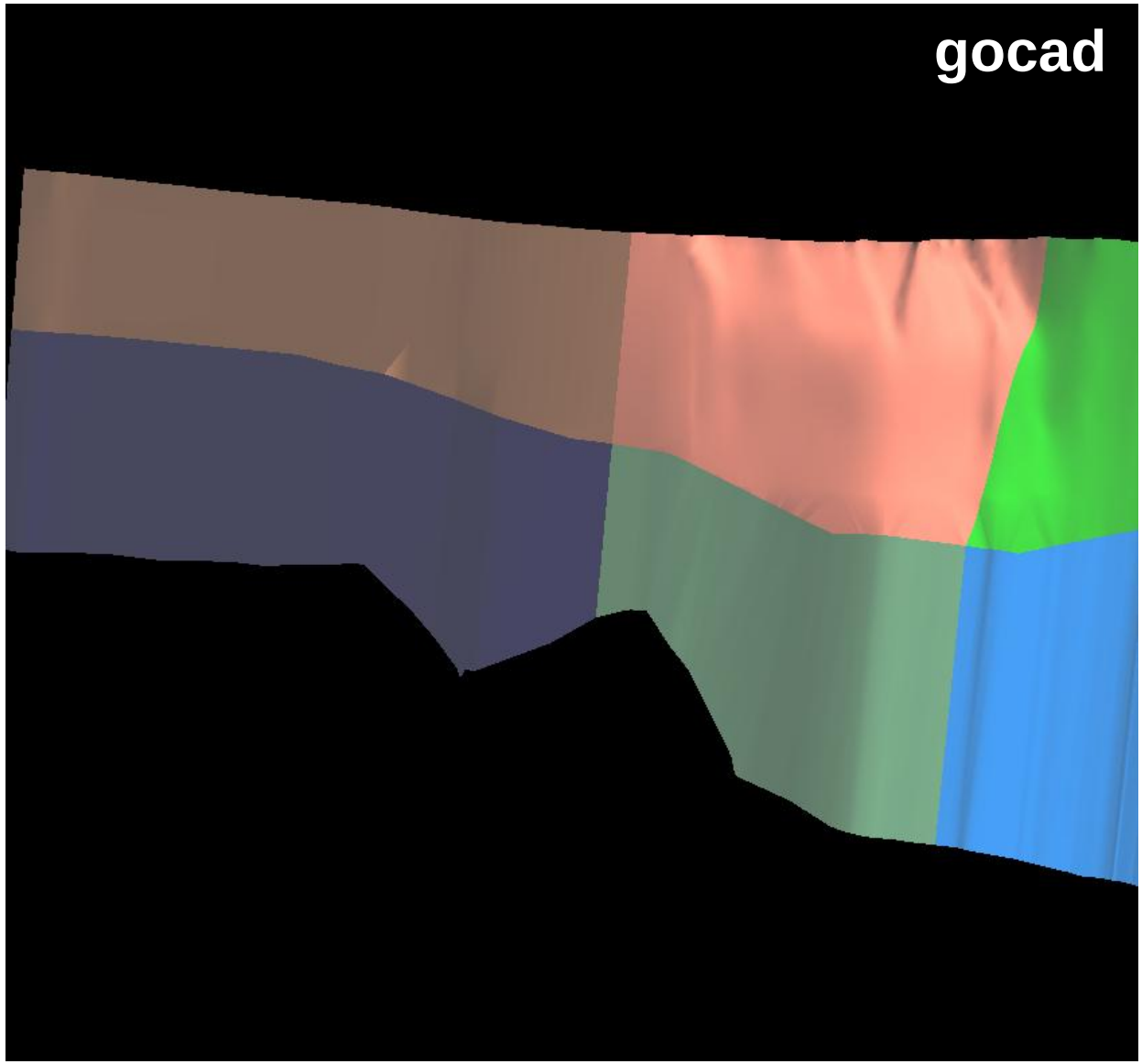
back to the Mojave CBM:

border is shown as a single TSurf, formed by merging separate TSurfs (create new TSurf from surfaces & merge parts with tolerances) defined in the Mojave CBM



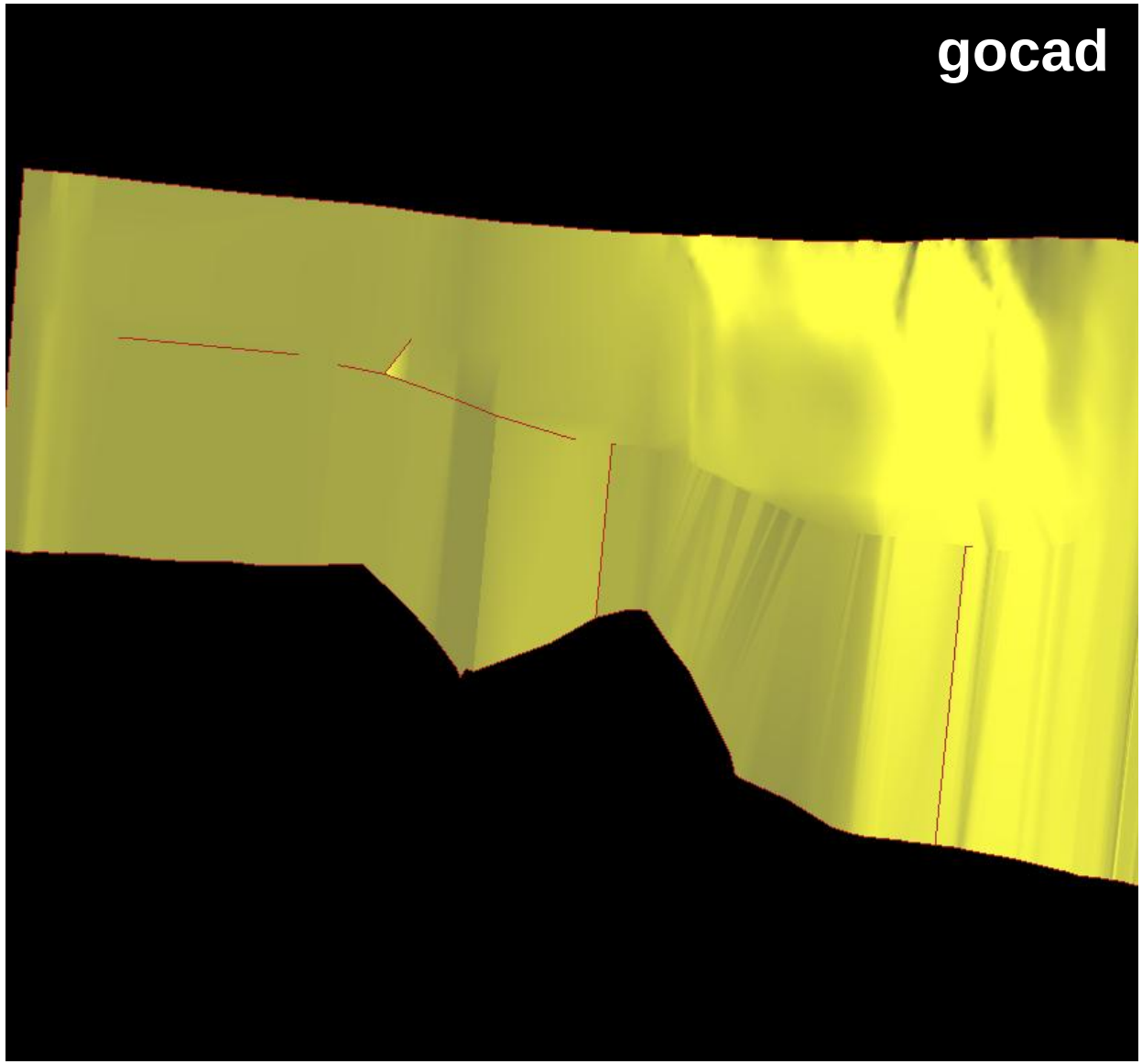
back to the Mojave CBM:

these are the original separate TSufs, probably separated in the first place from a single TSurf by a mutual cut among intersecting surfaces…

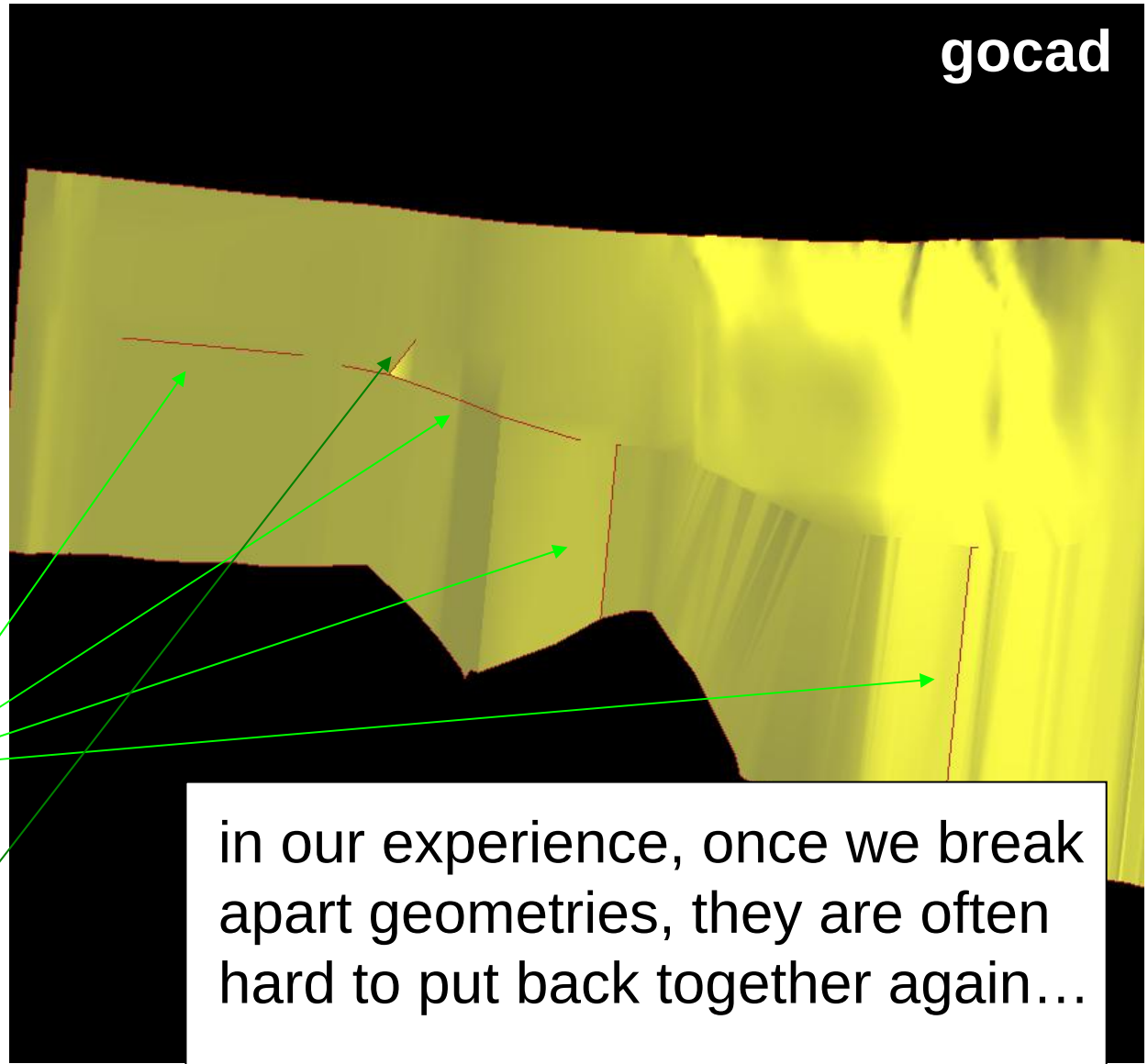


back to the Mojave CBM:

this is the single TSurf that was formed from all of the individual TSurfs, it should be continuous.



back to the Mojave CBM:



where the edges of the individual TSurfs do not meet within tolerances, the new single TSurf is non-continuous (represented by these red lines)

There is even an initial TSurf that was “cut” for some reason.

in our experience, once we break apart geometries, they are often hard to put back together again…

either in gocad or in cubit…

Remaining Work:

we might re-examine our choice of not using closed surfaces (CBM format?)…

we would like better control over the translation process, and we need to get a better handle on the cubit commands

we are building in automated error checking (and limited correcting) in the gocad ⇨ cubit phase

need to add material properties into ? ⇨ gocad ⇨ cubit

work on the ? ⇨ gocad step is ongoing…

cubit ⇨ pylith step still requires work, but it is well-defined…

scrapping cubit, and going with pure tet-mesh will avoid most of these problems

Remaining Questions:

How lost am I on gocad or cubit?

Can we get more control over importing triangle faceted surfaces into cubit?

Are the continuous, single surfaces in the CBM still available (in the CFM)?

How many people have I offended?

goals that we are not willing to sacrifice (at this point)

GOCAD: We would like a solution, such that we are always working with the highest level of the structural model, and we only form the solid model for each of the particular runs into a FE model.

MESHER: We do not want to remake the structural model within the mesher every time we generate a FE model.