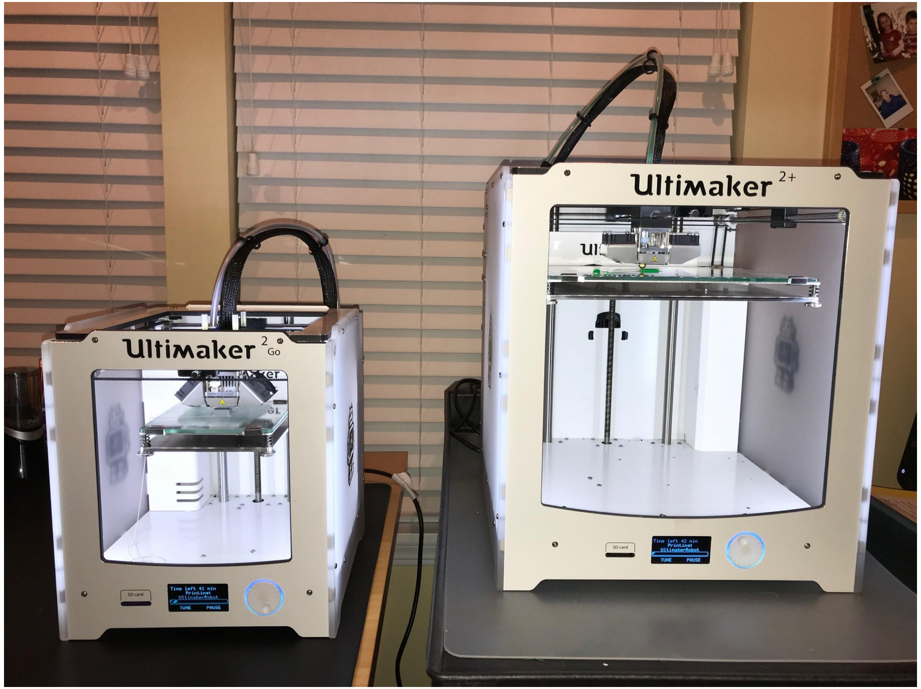
3D Printing Lessons in Multivariable Calculus

Kristen R. Schreck, D.A.
Saint Xavier University
Chicago, IL

ICTCM 2018

Ultimaker Education 3D Printing Pioneer



Teaching with 3D Printing

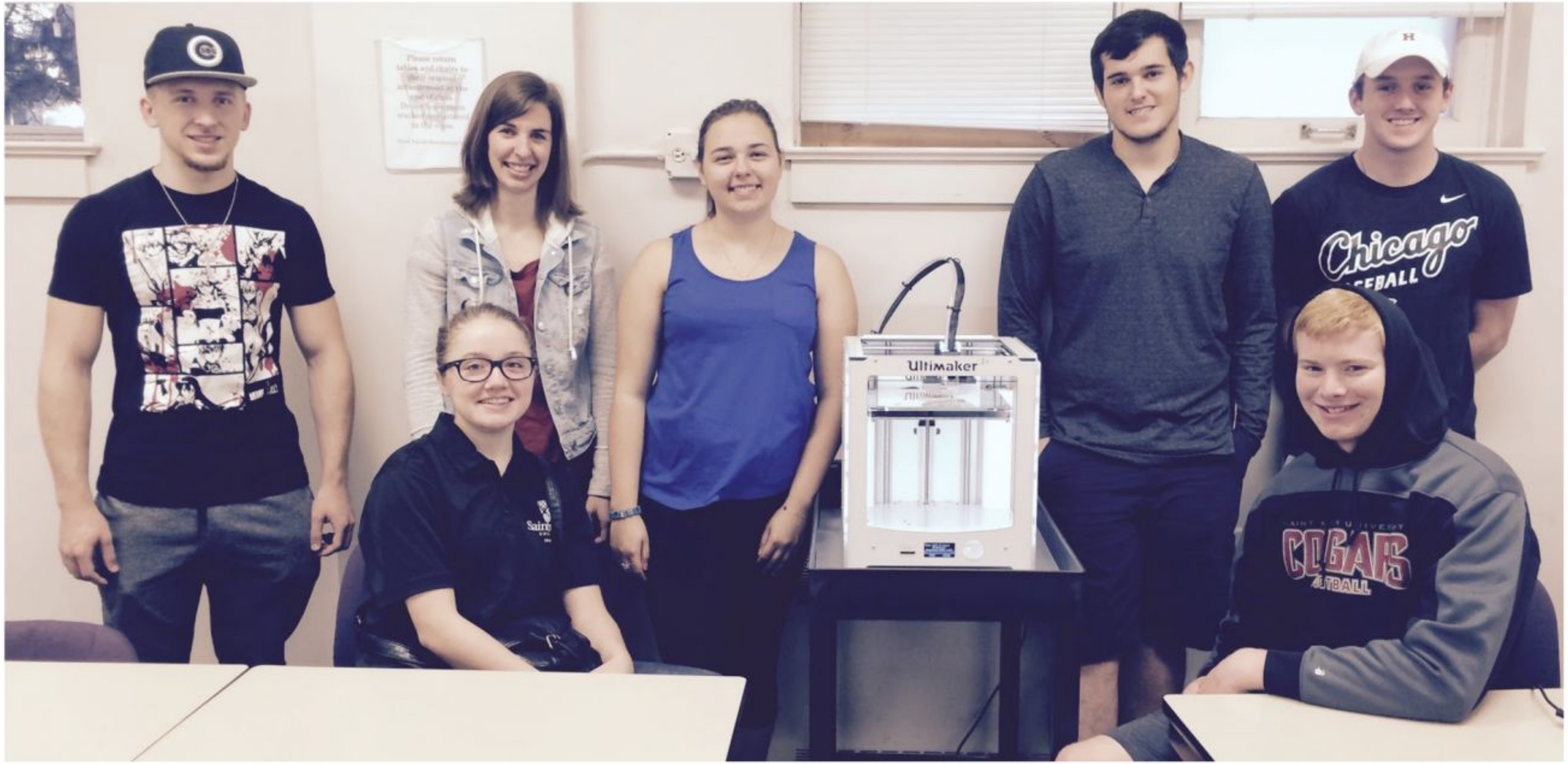
- Most of my students had no 3D modeling or printing experience

- Importance of iterative design process, creating prototypes

- Creativity, trial and error, refining analytical skills, building confidence

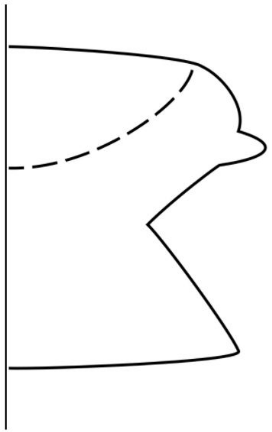
[Thingiverse: Nameplate Generator with OpenSCAD]

Inaugural Multivariable Calculus 3D Printing Class - Fall 2016



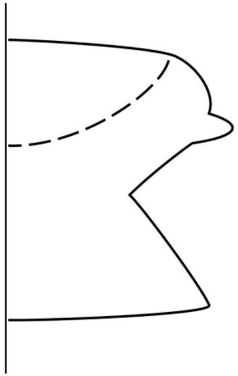
(Multivariable Calculus with 3D Printing)^2 - Fall 2017

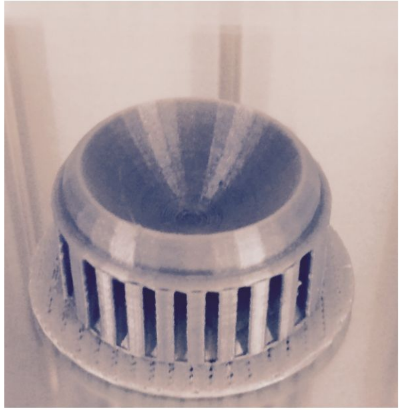
First Project - Tinkercad

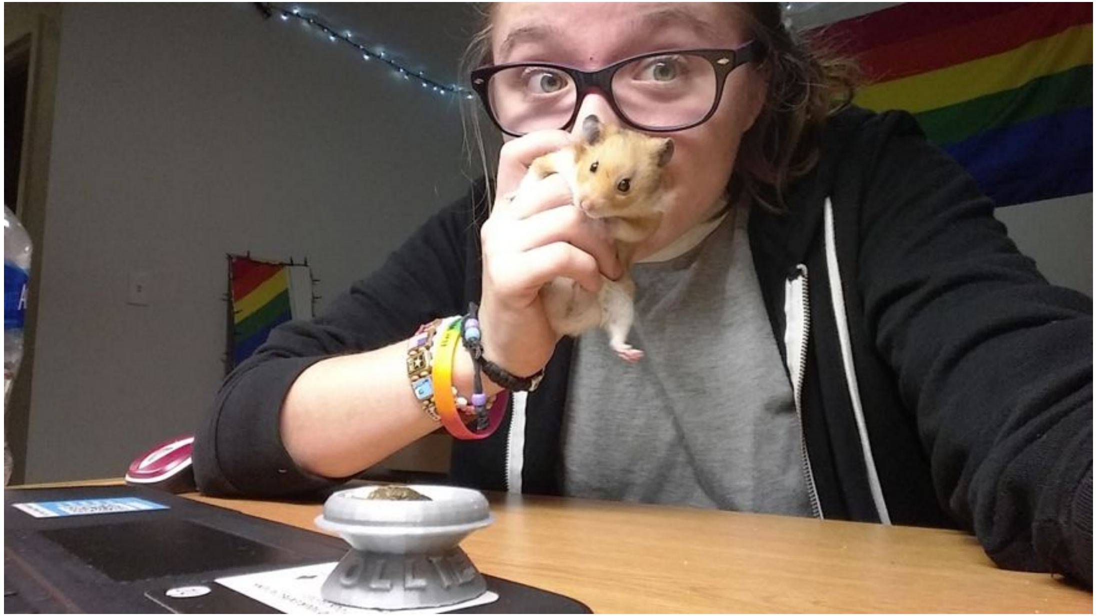


- Create an original surface of revolution using
  - paraboloid
  - ellipsoid
  - cylinder
  - cone
- Way to introduce 3D printing process steps
  - Design (& re-design)
  - Save as STL
  - Cura - 3D printer slicing software
  - 3D print

First 3D Designs & Prints - Tinkercad





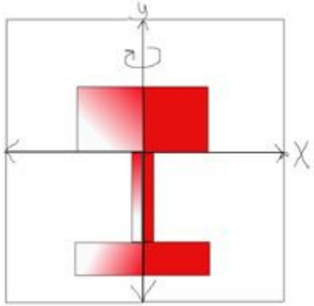


Rene with “Ollie” and his treat bowl

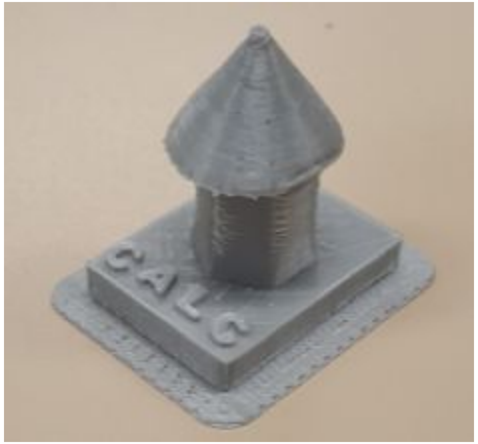
First Project - Tinkercad

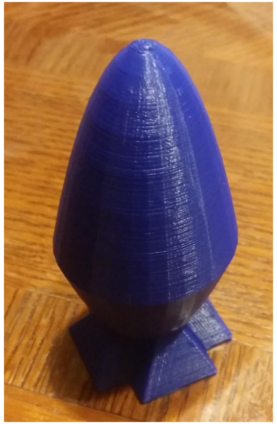
- Students document work
  - written report
  - video
  - class presentations

- Written reports
  - how models enhance mathematical understanding
  - 3D design & printing process details
  - include reflections on successes and pitfalls

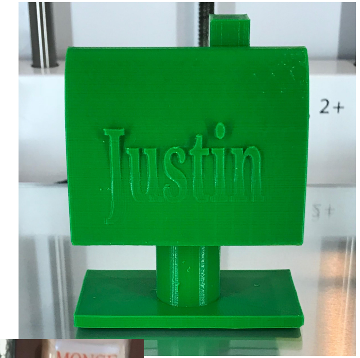


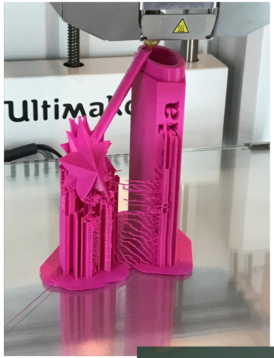
First 3D Designs & Prints - Tinkercad



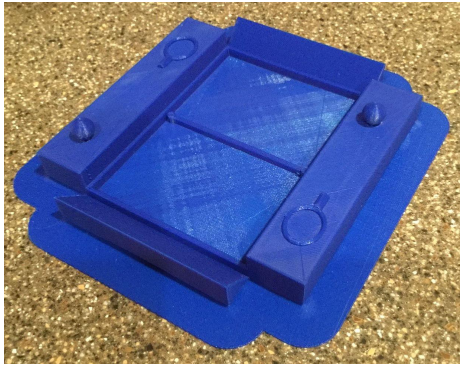


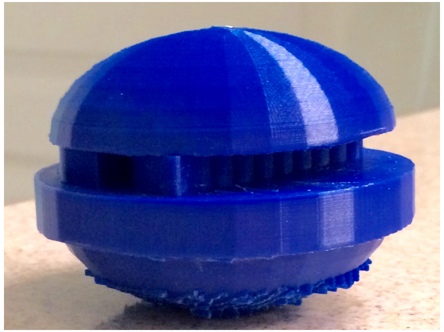
First 3D Designs & Prints

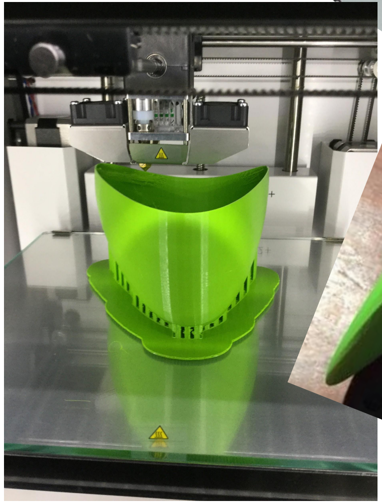


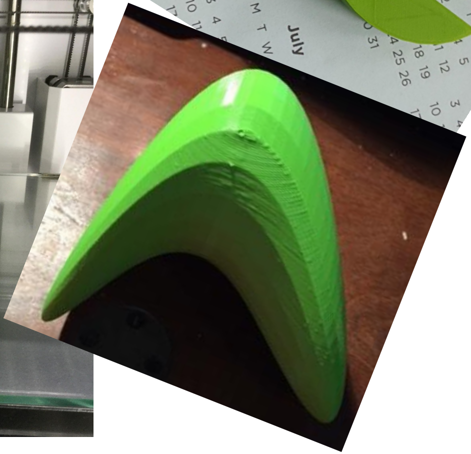


Other Imaginative Surfaces

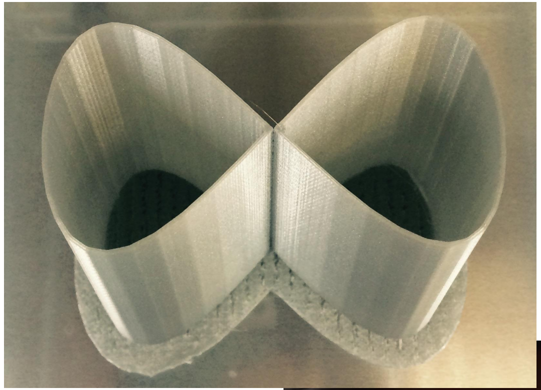


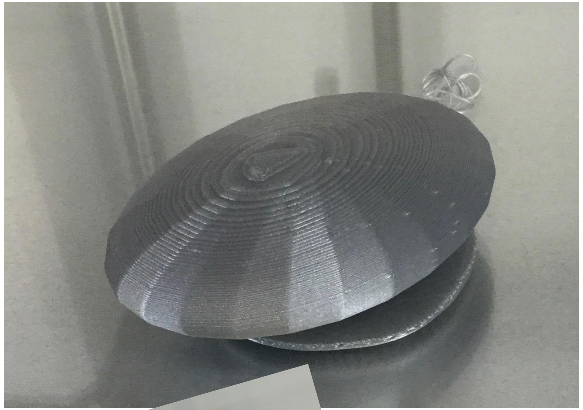


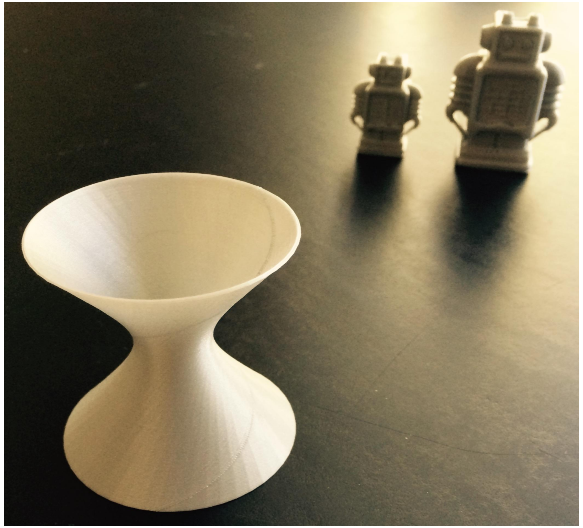


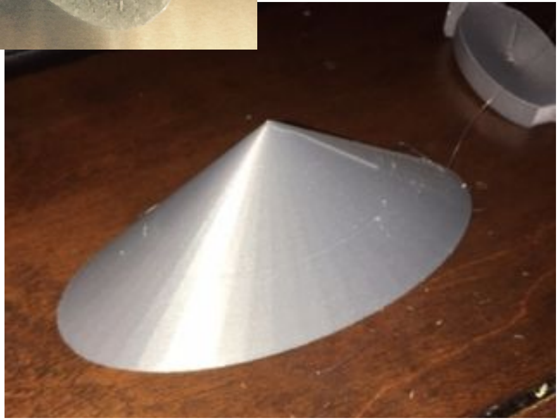


Second Project - Modeling Quadric Surfaces with Mathematica & Maple







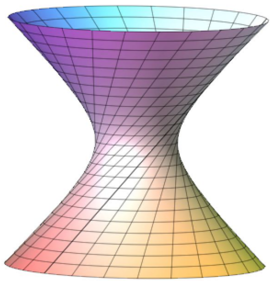


Quadric Surfaces: Maple & Mathematica

Hyperboloid of Revolution (Maple)

hyperboloid_revolution.mw

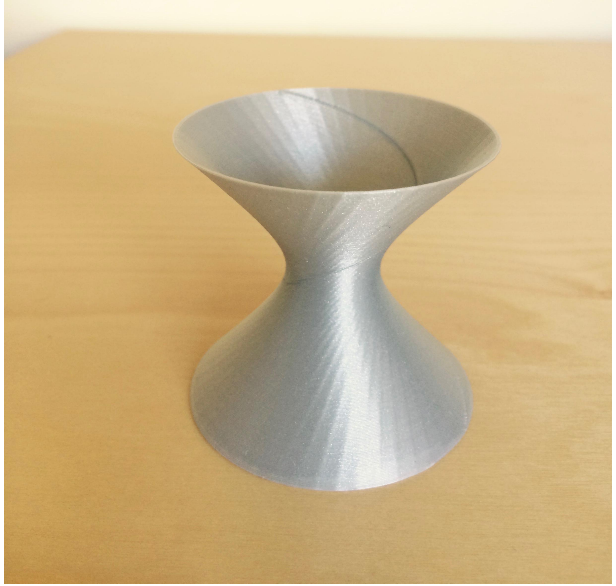
```
> restart :
  with(plots) :
  with(plottools) :
  with(VectorCalculus) :

> hyperboloid := ⟨cos(u) − v·sin(u), sin(u) + v·cos(u), v⟩ :
  plot3d(hyperboloid, u = 0..2·Pi, v = −3..3,
        style = surfacewireframe, lightmodel = light4, scaling = constrained,
        axes = none);
```



```
> hyperboloid := plot3d(⟨cos(u) − v·sin(u), sin(u) + v·cos(u), v⟩, u = 0..2·Pi, v = −3..3,
        style = surfacewireframe, lightmodel = light4, scaling = constrained,
        axes = none);
  myfile := FileTools:-JoinPath([currentdir( ), "myhyprev.stl"]);
  plottools[exportplot](myfile, hyperboloid);
                        hyperboloid := PLOT3D(...)
        myfile := "/Users/Kristen/Documents/3D with Maple 2016/myhyprev.stl"
                        230484
```

Second Project - Modeling Quadric Surfaces

- Each person (group) creates plots of assigned implicitly defined quadric surface
- uv-parameterizations were found to generate STL files
- MeshLab used to fix problems (or other surface chosen)
- Scaling adjusted, supports added, sliced in Cura, then 3D printed
- Documentation: mathematics of object, design specifics, problems, reflections

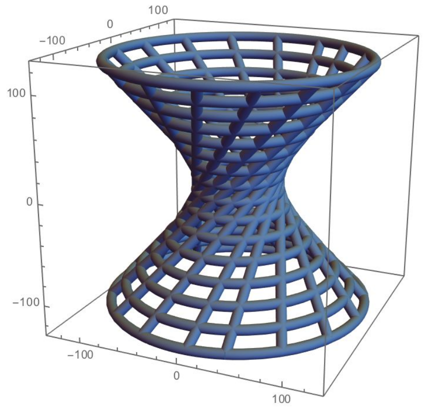


Quadric Surfaces - Hyperboloid of Revolution - Mathematica

Hyp_Rev_wireframe

In[64]:=
```
f[u_, v_] := {Cos[u] - v Sin[u], Sin[u] + v Cos[u], v};
scale = 40;
radius = 5;
numPoints = 24;
gridSteps = 5;
curvesU = Table[scale * f[u, i], {i, -3, 3, 2 / gridSteps}];
curvesV = Table[scale * f[j, v], {j, 0, 2 Pi, 2 / gridSteps}];
tubesU = ParametricPlot3D[curvesU, {u, 0, 2 Pi},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → All];
tubesV = ParametricPlot3D[curvesV, {v, -3, 3},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → All];
corners = Graphics3D[Table[Sphere[scale f[i, j], radius], {i, -3, 3, 2},
    {j, 0, 2 Pi, 2}], PlotPoints → numPoints];
output = Show[tubesU, tubesV]
Export["hyp_rev.stl", output]
```

Out[74]=

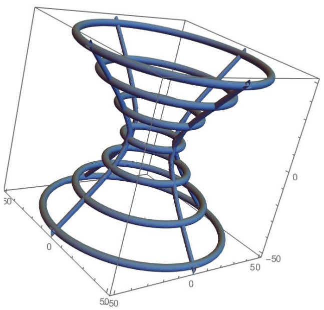


Out[75]= hyp_rev.stl

Hyp_Rev_contour3_wireframe

```
f[u_, v_] := {Cos[u] - v Sin[u], Sin[u] + v Cos[u], v};
Xp[t_] := {Sec[t], 0, Tan[t]};
Xn[t_] := {-Sec[t], 0, -Tan[t]};
Yp[t_] := {0, Sec[t], Tan[t]};
Yn[t_] := {0, -Sec[t], -Tan[t]};
scale = 16;
radius = 2;
radius1 = 1.2;
numPoints = 24
gridSteps = 2;
curvesZ = Table[scale * f[u, i], {i, -3, 3, 2 / gridSteps}];
tubesZ = ParametricPlot3D[curvesZ, {u, 0, 2 Pi},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → {-52.459, 52.459}];
tubesXp = ParametricPlot3D[scale * Xp[i], {i, -2 Pi, 2 Pi},
   PlotStyle → Tube[radius1, PlotPoints → numPoints], PlotRange → {-40, 40}];
tubesXn = ParametricPlot3D[scale * Xn[i], {i, -2 Pi, 2 Pi},
   PlotStyle → Tube[radius1, PlotPoints → numPoints], PlotRange → {-40, 40}];
tubesYp = ParametricPlot3D[scale * Yp[i], {i, -2 Pi, 2 Pi},
   PlotStyle → Tube[radius1, PlotPoints → numPoints], PlotRange → {-40, 40}];
tubesYn = ParametricPlot3D[scale * Yn[i], {i, -2 Pi, 2 Pi},
   PlotStyle → Tube[radius1, PlotPoints → numPoints], PlotRange → {-40, 40}];
output = Show[tubesZ, tubesXp, tubesXn, tubesYp, tubesYn, tubesU]
Export["hyprev_contour_best.stl", output]
```

24



hyprev_contour_best.stl

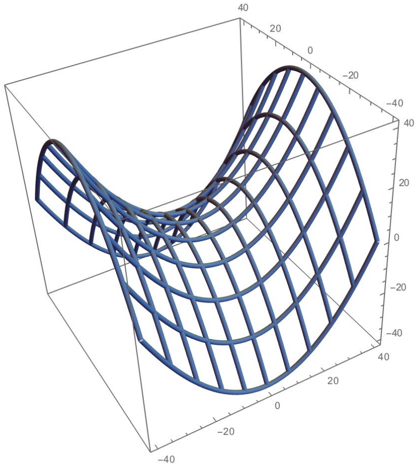
Hyperboloids of Revolution

Quadric Surfaces - Saddle Surface - Mathematica

Saddle_wireframe

```
In[76]:= f[u_, v_] := {u, v, u^2 - v^2};
scale = 40;
radius = 0.75;
numPoints = 24;
gridSteps = 10;
curvesU = Table[scale * f[u, i], {i, -1, 1, 2 / gridSteps}];
curvesV = Table[scale * f[j, v], {j, -1, 1, 2 / gridSteps}];
tubesU = ParametricPlot3D[curvesU, {u, -1, 1},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → All];
tubesV = ParametricPlot3D[curvesV, {v, -1, 1},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → All];
corners = Graphics3D[Table[Sphere[scale f[i, j], radius], {i, -1, 1, 2},
    {j, -1, 1, 2}], PlotPoints → numPoints];
output = Show[tubesU, tubesV, corners]
Export["saddle.stl", output]
```

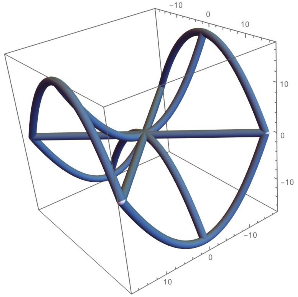
Out[86]=



Out[87]= saddle.stl

Saddle_contours

```
f[u_, v_] := {u, v, u^2 - v^2};
Lp[t_] := {t, t, 0};
Ln[t_] := {t, -t, 0};
X[t_] := {0, t, -t^2};
Y[t_] := {t, 0, t^2};
Zp[x_, y_] := {x, y, 1 - x^2 + y^2};
scale = 16;
radius = 0.75;
numPoints = 24;
gridSteps = 2;
gridSteps1 = 5;
curvesU = Table[scale * f[u, i], {i, -1, 1, 2 / gridSteps}];
curvesV = Table[scale * f[j, v], {j, -1, 1, 2 / gridSteps}];
tubesU = ParametricPlot3D[curvesU, {u, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints],
   PlotRange → All];
tubesV = ParametricPlot3D[curvesV, {v, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints],
   PlotRange → All];
tubesLp = ParametricPlot3D[scale * Lp[i], {i, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints]
   PlotRange → All];
tubesLn = ParametricPlot3D[scale * Ln[i], {i, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints]
   PlotRange → All];
tubesX = ParametricPlot3D[scale * X[i], {i, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints],
   PlotRange → All];
tubesY = ParametricPlot3D[scale * Y[i], {i, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints],
   PlotRange → All];
curvesZp = Table[scale * Zp[i, y], {i, -1, 1, 2 / gridSteps}];
tubesZp = ParametricPlot3D[curvesZp, {y, -1, 1}, PlotStyle → Tube[radius, PlotPoints → numPoints],
   PlotRange → All];
corners = Graphics3D[Table[Sphere[scale f[i, j], radius], {i, -1, 1, 2}, {j, -1, 1, 2}],
   PlotPoints → numPoints];
output = Show[tubesV, tubesU, tubesLp, tubesLn, tubesX, tubesY, corners]
Export["saddle_contours.stl", output]
```



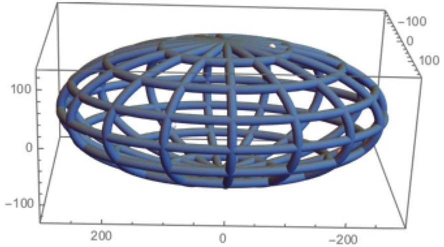
saddle_contours.stl

Saddle Surfaces

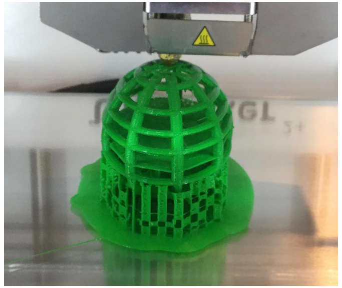
Ellipsoid, Sphere, Paraboloid

ellipsoid

```
f[u_, v_] := {7 Cos[u] Sin[v], 4 Sin[u]  Sin[v], 3 Cos[v]};
scale = 40;
radius = 8;
numPoints = 24;
gridSteps = 5;
curvesU = Table[scale * f[u, i], {i, 0, Pi, 2 / gridSteps}];
curvesV = Table[scale * f[j, v], {j, 0, 2 Pi, 2 / gridSteps}];
tubesU = ParametricPlot3D[curvesU, {u, 0, 2 Pi},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → All];
tubesV = ParametricPlot3D[curvesV,  {v, 0, Pi},
   PlotStyle → Tube[radius, PlotPoints → numPoints], PlotRange → All];
corners = Graphics3D[Table[Sphere[scale f[i, j], radius], {i, 0, Pi, 2}, {j, 0, 2 Pi, 2}],
   PlotPoints → numPoints];
output = Show[tubesU, tubesV, corners]
Export["ellipsoid.stl", output]
```




```
ellipsoid.stl
```

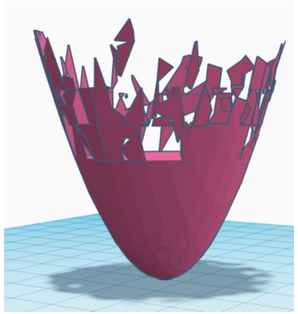


Challenges for Students

$$f(x,y)=\frac{x^2}{a^2}+\frac{y^2}{b^2}$$

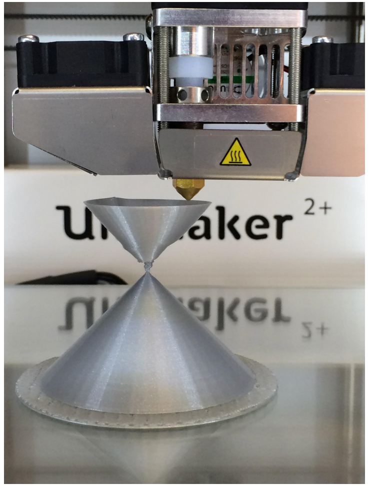
$$f(u,v)=\left(a\sqrt{u/h}\cos(v),a\sqrt{u/h}\sin(v),u\right)$$

- **Math**
  - Implicit form to *u*, *v* parameterization for quadric surfaces
  - Code to print wireframe vs. solid surfaces



$$f(u,v)=\left(u,v,\frac{u^2}{a^2}+\frac{v^2}{b^2}\right)$$

- **3D Printing**
  - Determining best orientation of object
  - When to use supports
  - Cura settings
  - Fixing problems with triangular meshes



Qlone App - 3D Scanning

Very Entertaining Student 3D Printing Videos

- Hamster Dish
- A Little House Music
- Double Helix
- Personalized Cup
- Mailbox
- Goblet
- Make-up

Student Perspectives on 3D Printing

- “My experiences with 3D printing in this course have been phenomenal. I have been able to create designs that I thought of, but also create designs that were based off functions studied in the course. This has elevated my learning of the material.”

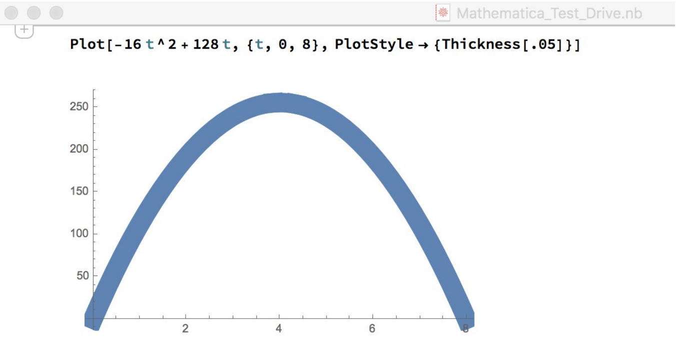
- “There are hiccups in math, and 3D printing is no exception. Troubleshooting problems, making mistakes, and ultimately fixing them is a crucial part of learning that 3D printing let me explore within math and using the software.”

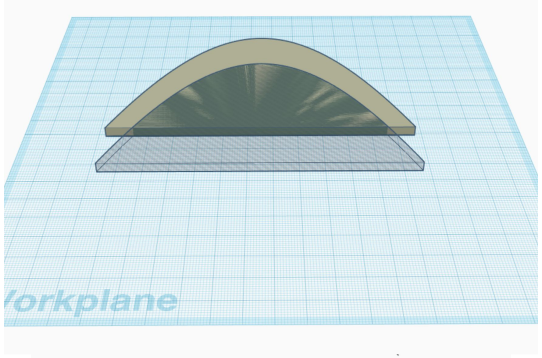
- “It’s fun to make objects, but the fact that we now know how the objects are made with our knowledge of implicit functions and parameterizations makes it that much better. As a future educator, this is what I want to show my students: Math is everywhere and you will use it.”

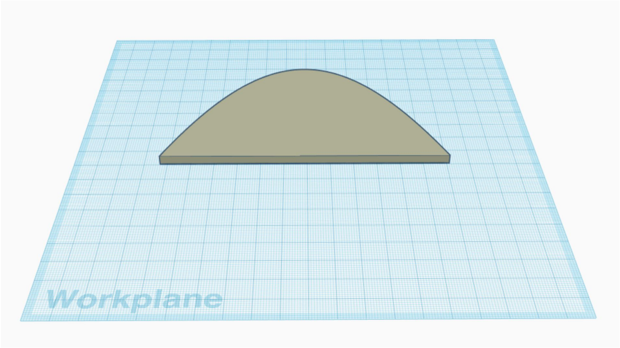
Teacher Perspectives on 3D Printing

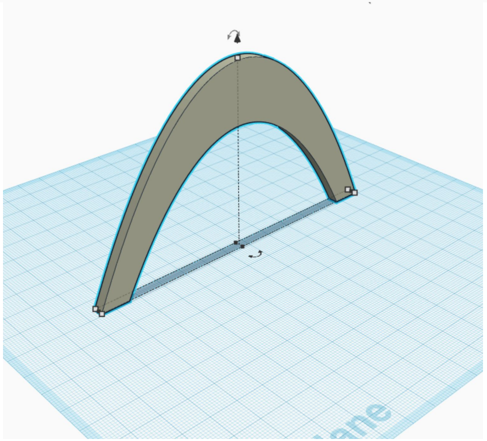
- Joy of watching students see a mathematical object *they designed* 3D printed for the first time
- Students need time to create 3D designs (they think about it a lot!) and get to know the software on their own
- Student writing component: answers too brief, mathematical description not in-depth
- Reminder to students: PLA filament is not food-grade
- Extra time is need to edit objects to obtain clean 3D prints and remove supports (have the right tools)
- Next ideas: Activity - surfaces with level curves; Volumes - intersections of surfaces with iterated integrals

Calculus I - Illustrating Theorems 2D to 3D

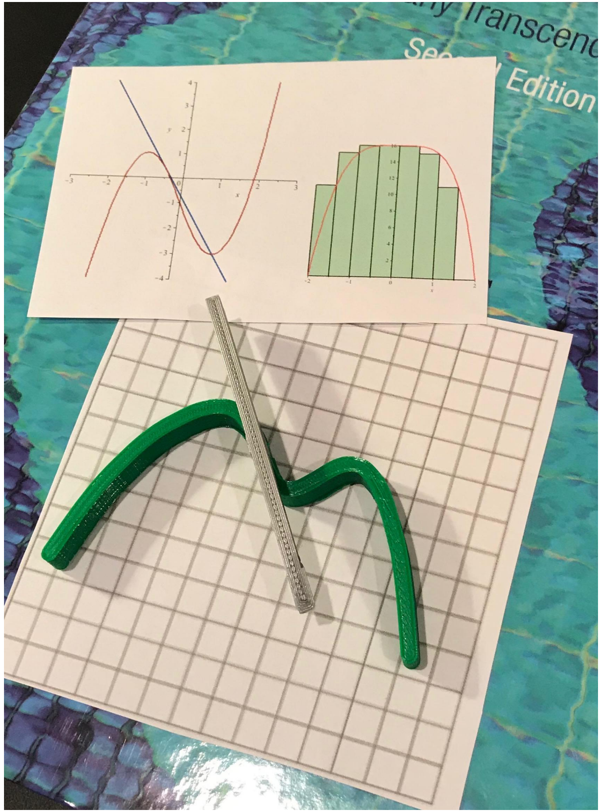




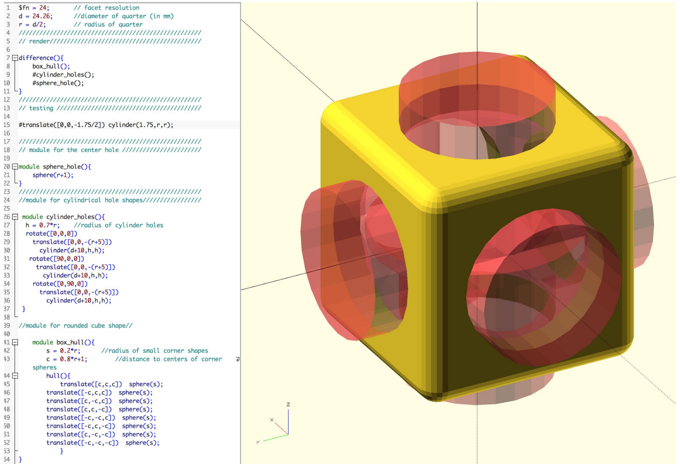


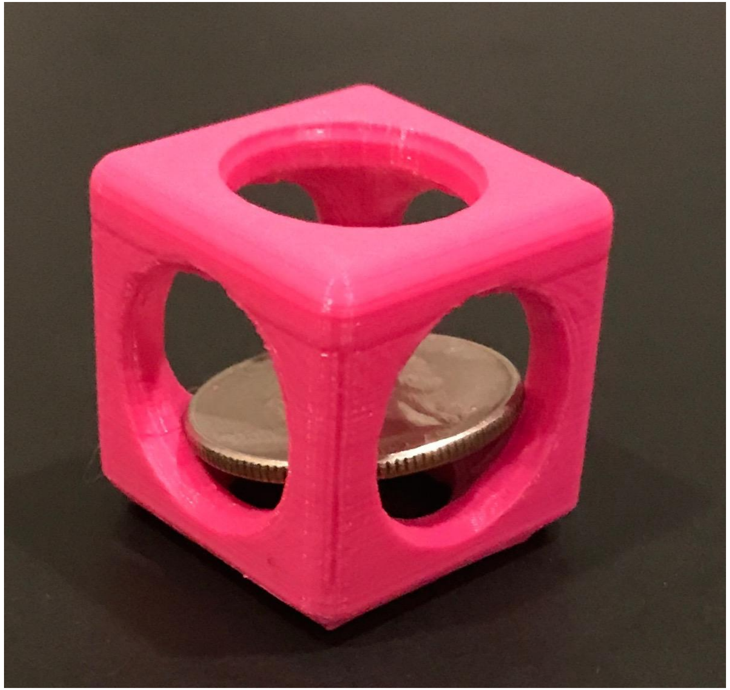


Calculus I - Illustrating Theorems 2D to 3D



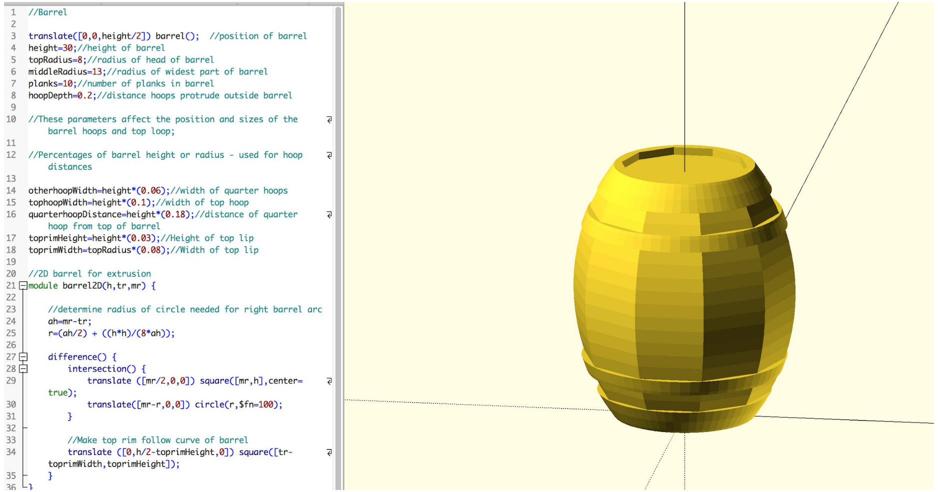
Modern Geometry - Constructive Solid Geometry Quarter Trap - OpenSCAD





- Inspiration: [MakerHome: Day 314]
- My Lesson: [MathIn3D]

Senior Seminar - Advanced LaGrange Multipliers - Business Applications Package Design & Kepler's Wine Barrel Problem - OpenSCAD

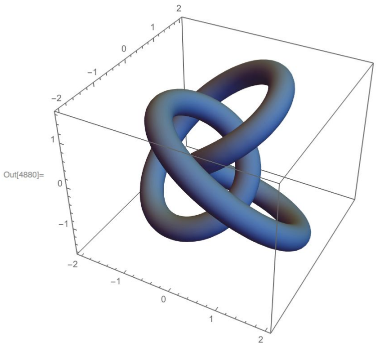


- [Kepler's Wine Barrel Problem]
- [The PuzzleGeek]

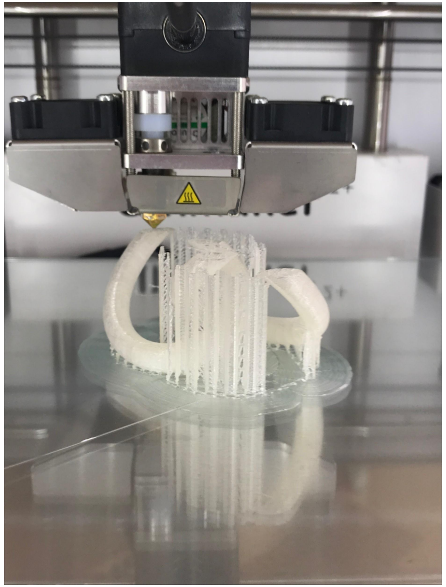
Topology: The Rocking Knot (Mathematica)

rocking_knot

```
In[4878]:= a = .8
b = Sqrt[1 - a^2]
(*a and b have to be numbers with a^2+b^2=1*)
ParametricPlot3D[{a*Cos[3 t] / (1 - b*Sin[2 t]), a*Sin[3 t] / (1 - b*Sin[2 t]),
  1.8*b*Cos[2 t] / (1 - b*Sin[2 t])}, {t, 0, 2 Pi + .01},
 PlotStyle → Tube[0.25, PlotPoints → 24], PlotRange → All]
Export["tritangentless_thick.stl", %]

Out[4878]= 0.8

Out[4879]= 0.6

Out[4880]=
```

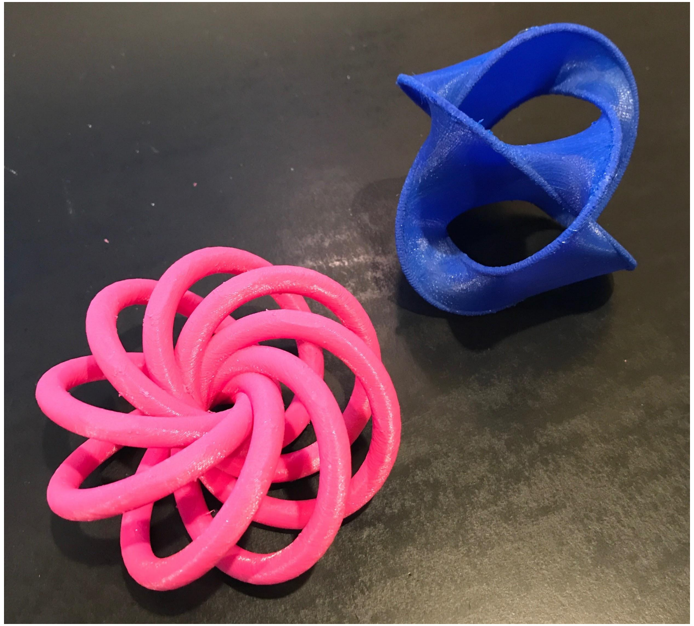


```
Out[4881]= tritangentless_thick.stl
```

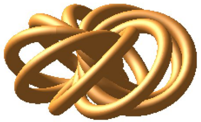


Knot parameterization:
[Laura Taalman's Makerhome blog: Day 110 - the Rocking Knot]

Topology: Torus Knot (Maple) & Seifert Surface for the Borromean Rings



```
> with(algcurves) :
> printlevel := 2 :
> plot_knot(y^8 − x^8, x, y, color = gold, numpoints = 100, tubepoints = 100, radius = .2, axes = none);
                                   Number of branches:, 8
```



```
> TorusKnot := plot_knot(y^8 − x^8, x, y, color = gold, numpoints = 100, tubepoints = 100, radius = 0.2, axes = none);
                                   Number of branches:, 8
                          [Length of output exceeds limit of 1000000]
> myfile := FileTools:-JoinPath([currentdir( ), "TorusKnot8.stl"]);
                  myfile := "/Users/Kristen/Desktop/Ks 3D Prints 2017/TorusKnot8.stl"
> plottools[exportplot](myfile, TorusKnot);
```

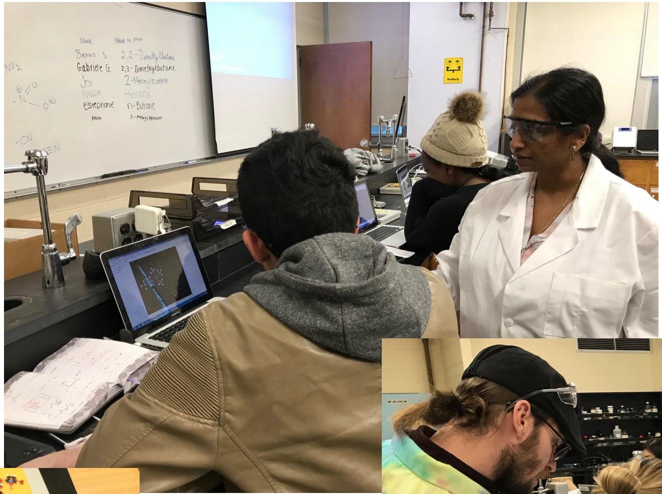
Seifert Surface help page:
[MakerHome: Day 285]

Southwest Chicago Math Teachers' Circle - Hexaflexagons

To make these hexaflexagons, I modified the OpenSCAD code to create my own version of https://www.thingiverse.com/thing:1534607

Biochemical Molecules - Design, Model, 3D Print

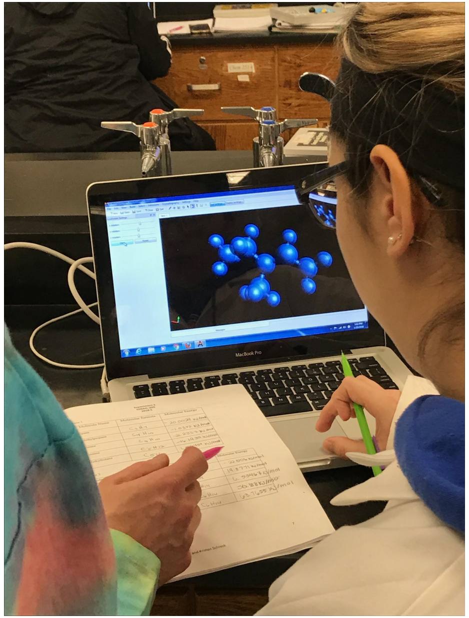
- **Dr. Sharada Buddha**
  SXU Associate Professor of Chemistry
- **Curtis Feipel**
  SXU Biology Major and Chemistry Minor
- Inspiration:
  Dr. W. Tandy Grubbs
  Stetson University
  [3D Printable Molecular Models]()



Biochemical Molecules - Design, Model, 3D Print

Avogadro

- molecular editor and visualization tool



Biochemical Molecules

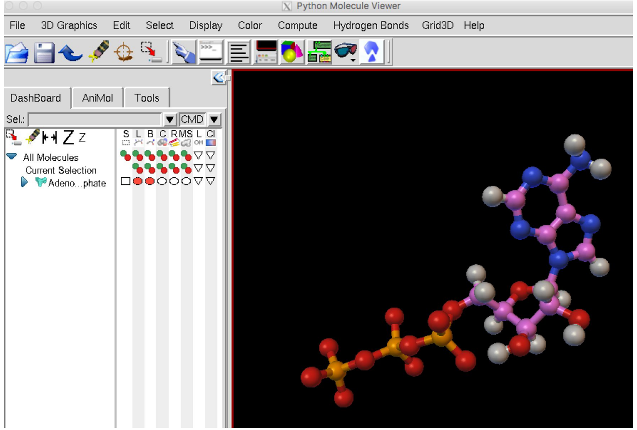
- Cyclo-propane
  -hexane
  -butane
  -pentane

- Hexane
  Dimethylcyclopentane
  Dimethylbutane
  N-butane

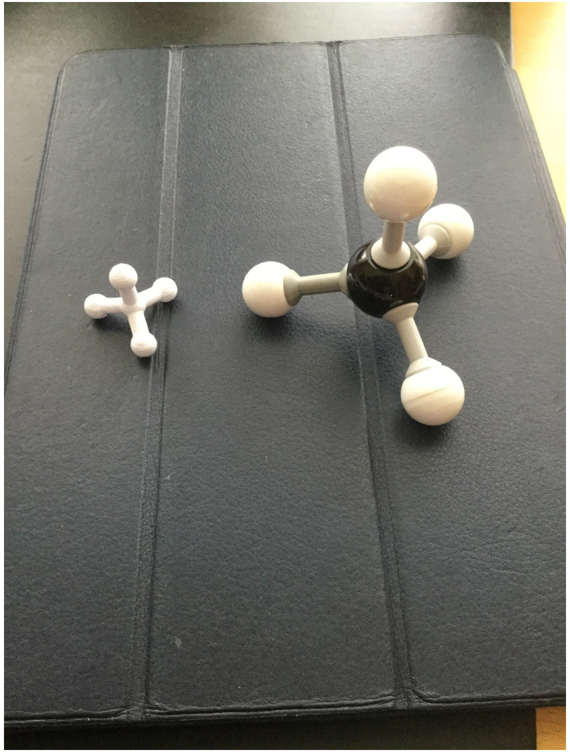
- Adenosine triphosphate (ATP)
  Glucose

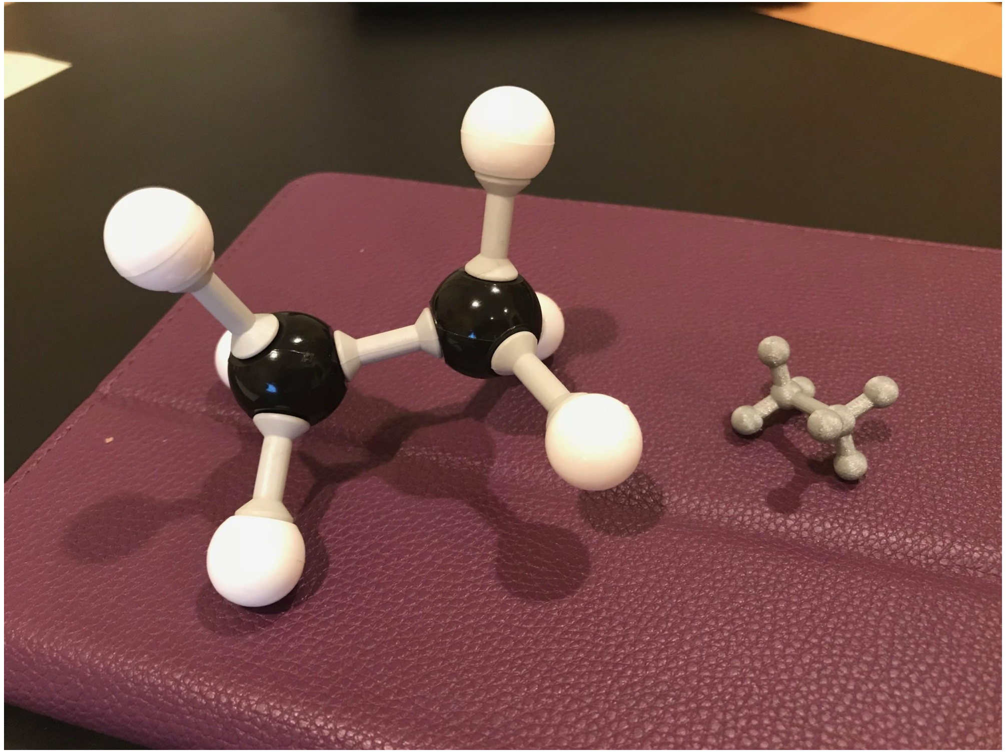
Python Molecular Viewer to STL file for 3D printing

- converts Avogadro chemical model to STL file for 3D printing

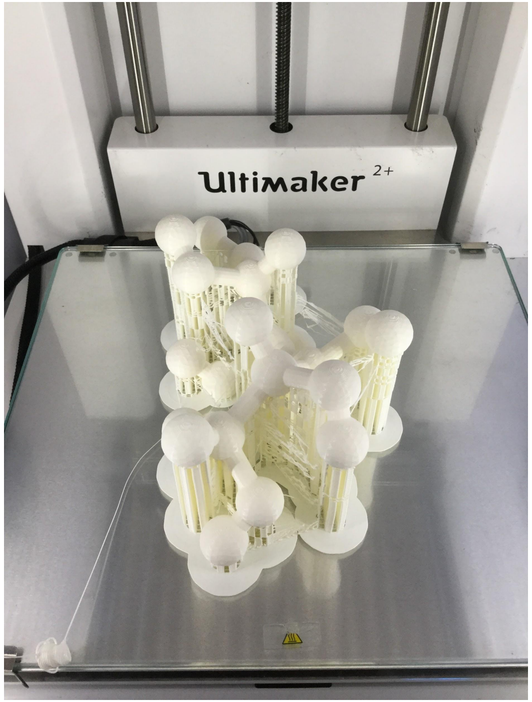


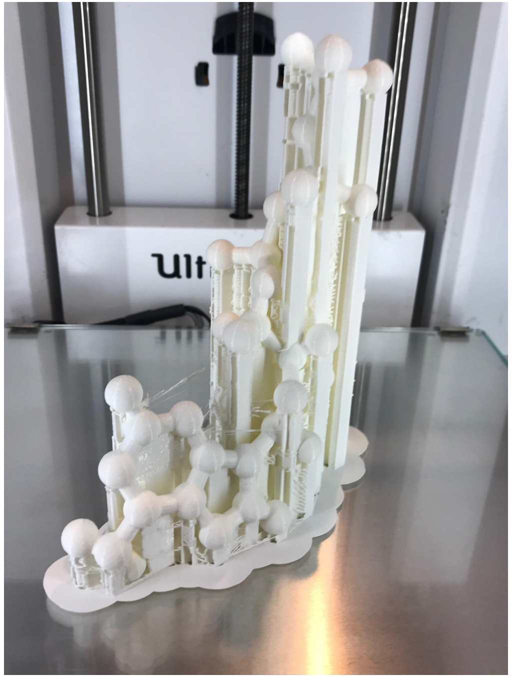
Biochemical Molecules - Design, Model, 3D Print





Biochemical Molecules - Design, Model, 3D Print





Sneak Peek: Ultimaker Education Pioneer Project

I am working with three fellow Pioneers on a top-secret project!

- **Greg Kent**, Technology Coordinator at Kailua Elementary School, Hawaii
- **Alex Larson**, Career and Technical Education teacher at Palatine High School, Illinois
- **Brian Wetzel**, Computer Technology teacher at Centerburg High School, Columbus, Ohio.

We will be presenting the results of our collaboration at Construct3D 2018 at Georgia Tech later this year.

Senior Seminar Spring 2018: Visualizing Hyperbolic Geometry

http://www.segerman.org/

My Blog Posts and Publications related to 3D design and printing in Math:

- Preparing to Teach with 3D Printing
- Out of the Box - Ultimaker 2+ First Impressions
- Our 3D Printing Journey in Multivariable Calculus
- Monge's Legacy of Descriptive and Differential Geometry

My 3D Printing Lessons

- Quadric Surfaces with Maple
- An Imaginative Surface using Concepts from Multivariable Calculus
- Surface of Revolution using Tinkercad

3D Design Software Used

CAD & Modeling

- Tinkercad (free)
- OpenSCAD (free)
- Morphi (nominal $)
- Blender (free)

Mathematical

- Mathematica (link to 3D Printing) ($)
- Maple (link 3D Printing) ($)

Processing & Editing 3D files

- MeshLab (free)

Experimenting with 3D Scanning

- Qlone (free)

3D Printing in Mathematics - The Real Pioneers

- **Laura Taalman/mathgrrl** (James Madison University)
    - http://mathgrrl.com/hacktastic/home/

- **Elizabeth Denne** (Washington and Lee University)
    - http://home.wlu.edu/~dennee/math_vis.html#Instructions
    - http://mathvis.academic.wlu.edu/

- **Christopher Hanusa** (Queens College)
    - https://qcpages.qc.cuny.edu/~chanusa/mathematica/

3D Printing in Mathematics - The Real Pioneers

- **Henry Segerman** (Oklahoma State University)
    - http://www.segerman.org/
- **Vi Hart**
    - http://vihart.com/
- **John Zweck** (University of Texas at Dallas)
    - https://www.utdallas.edu/~jwz120030/3DPrintedModelsForCalcIII/

Thank you!