

# From grids to surfaces and back again

## OEGrid and OESpicoli Introduction

Brian Cole  
CUP IX



# Interpreting the Zap results

- Associating induced charge with atoms
  - Per atom induced charge contribution
- Analyzing induced charge
  - Choosing contours
  - Analyze the contours



# Per atom induced charge

- Make a molecular Gaussian grid
- For each atom in molecule
  - Make a Gaussian grid for the atom
  - Divide the atom grid by the molecule grid
  - Multiply by the induced charge grid
  - Sum over all grid points
  - Multiply by the cell volume
  - Assign to the atom

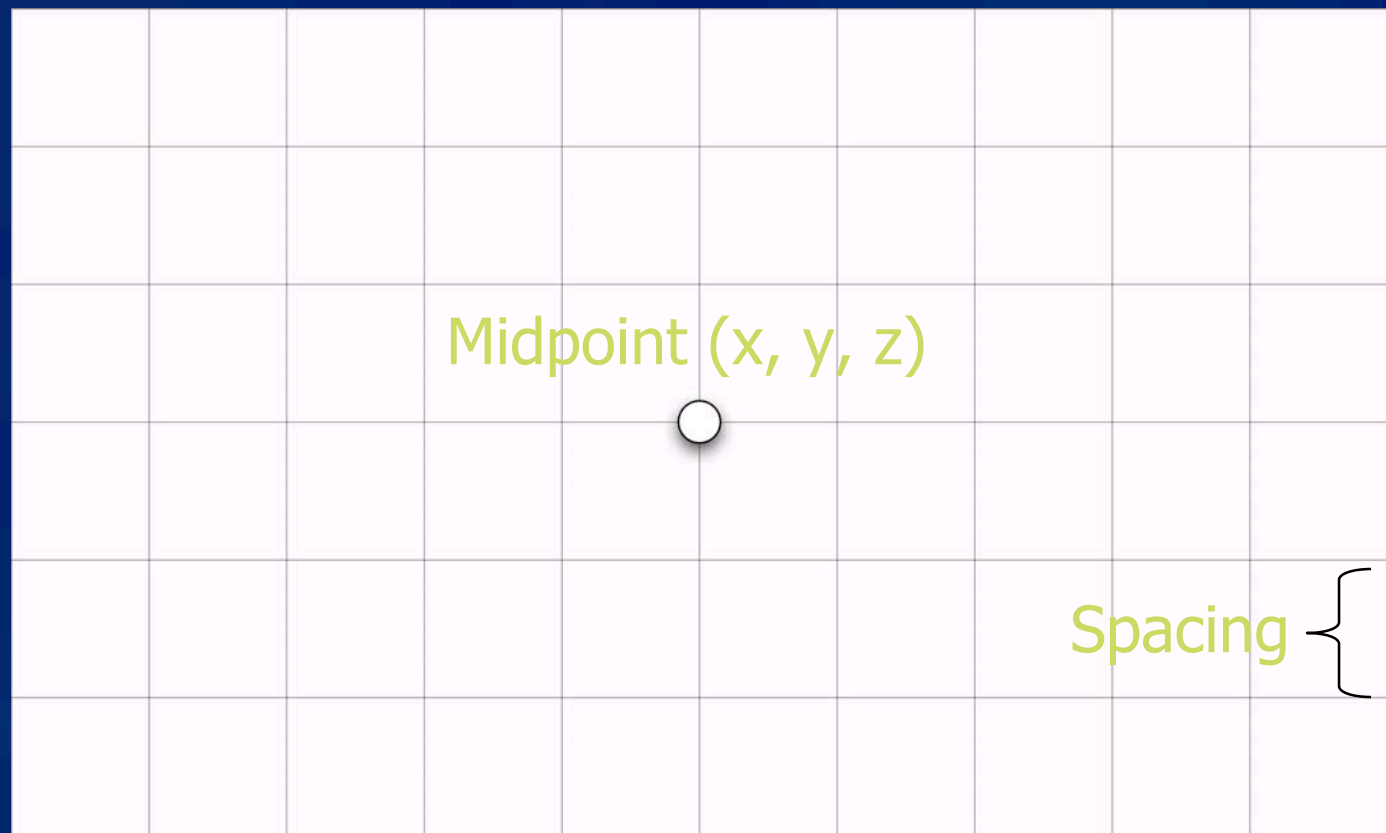
# OEGrid

- Comes with OEChem
- Supports lattices as first class objects
  - OEScalarGrid
  - OESkewGrid
- Features
  - IO from several formats (gridbabel)
  - Attach as generic data to molecules
  - Regularization
  - Masking
  - Interpolation
  - The sky is the limit (or the amount of memory you have)



# Grid Fundamentals

Dimensions



# Copying Grid Geometry

```
def CopyGridGeometry(grid):  
    newgrid = OEScalarGrid()  
    newgrid.SetDim(*grid.GetDim())  
    newgrid.SetMid(*grid.GetMid())  
    newgrid.SetSpacing(grid.GetSpacing())  
    return newgrid
```



# What are grids?

- Lattices of values

0.01	0.18	0.82	1.21	8.30	12.11	5.64	0.84	0.04	0.00
0.06	0.93	4.33	6.43	11.89	17.36	8.08	1.20	0.11	0.01
0.11	1.57	7.33	10.89	17.69	12.12	4.94	3.33	0.71	0.05
0.06	0.85	3.96	5.88	12.05	17.59	9.88	6.65	1.43	0.10
0.01	0.15	0.66	4.45	9.56	6.55	6.30	4.24	0.91	0.06
0.00	0.02	0.50	3.35	7.20	4.93	1.28	0.86	0.18	0.01

CH<sub>3</sub>  
O



# Gaussian Grids

```
def MakeGaussianGridOfSameGeometry(grid, mol):  
    molgrd = OEScalarGrid()  
    OEMakeMolecularGaussianGrid(molgrd, mol, grid.GetSpacing())  
  
    fullmolgrd = CopyGridGeometry(grid)  
    OEInterpolateBetweenGrids(fullmolgrd, molgrd)  
  
    return fullmolgrd
```



# Gaussian grids for every atom

```
def GridToAtom(valuegrid, totalgrid, atom):  
    assert valuegrid.GetSize() == totalgrid.GetSize()  
  
    newmol = OEGraphMol()  
    newmol.NewAtom(atom)  
  
    atmgrd = MakeGaussianGridOfSameGeometry(totalgrid, newmol)
```



# Mapping grid values to atoms

```
# OEDivideScalarGrid(atmgrd, totalgrid)
for i in xrange(atmgrd.GetSize()):
    totval = totalgrid[i]
    if totval != 0.0:
        atmgrd[i] /= totval
    else:
        atmgrd[i] = 0.0
```

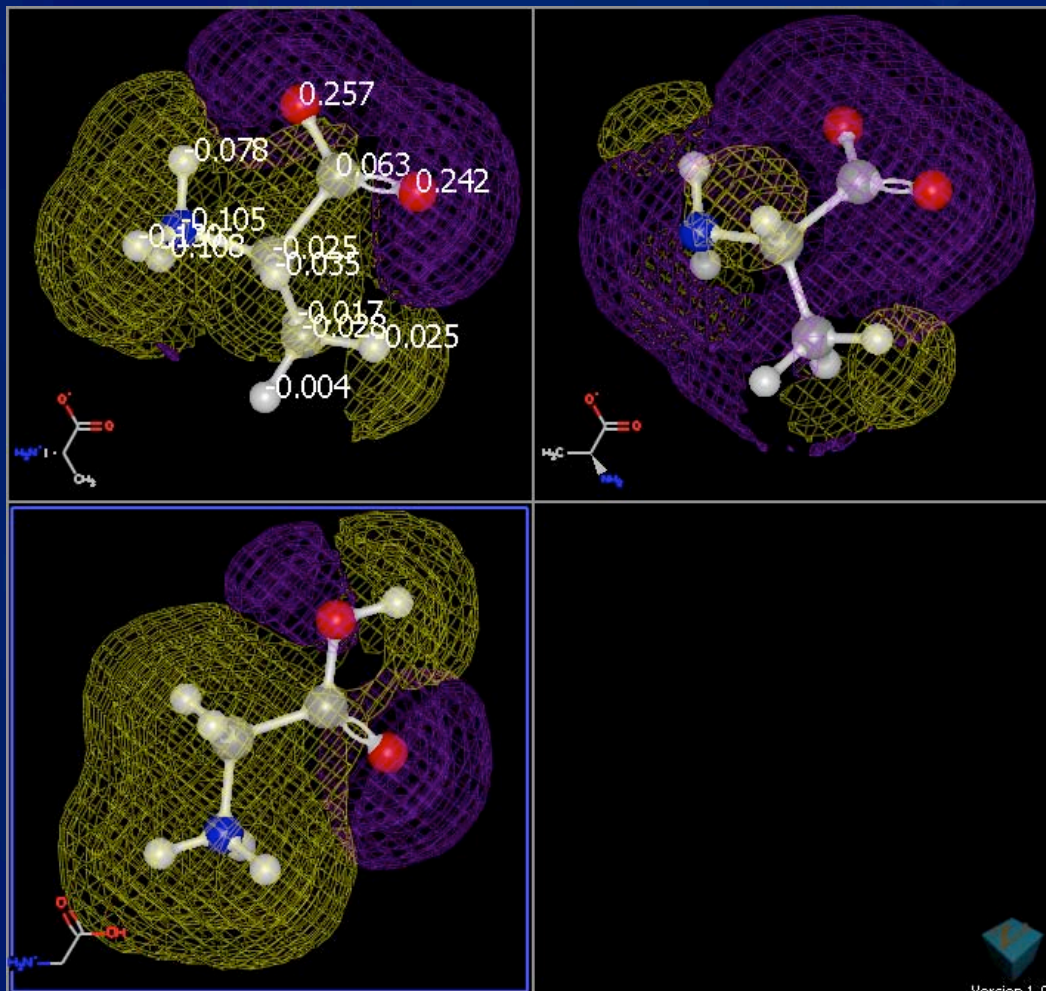
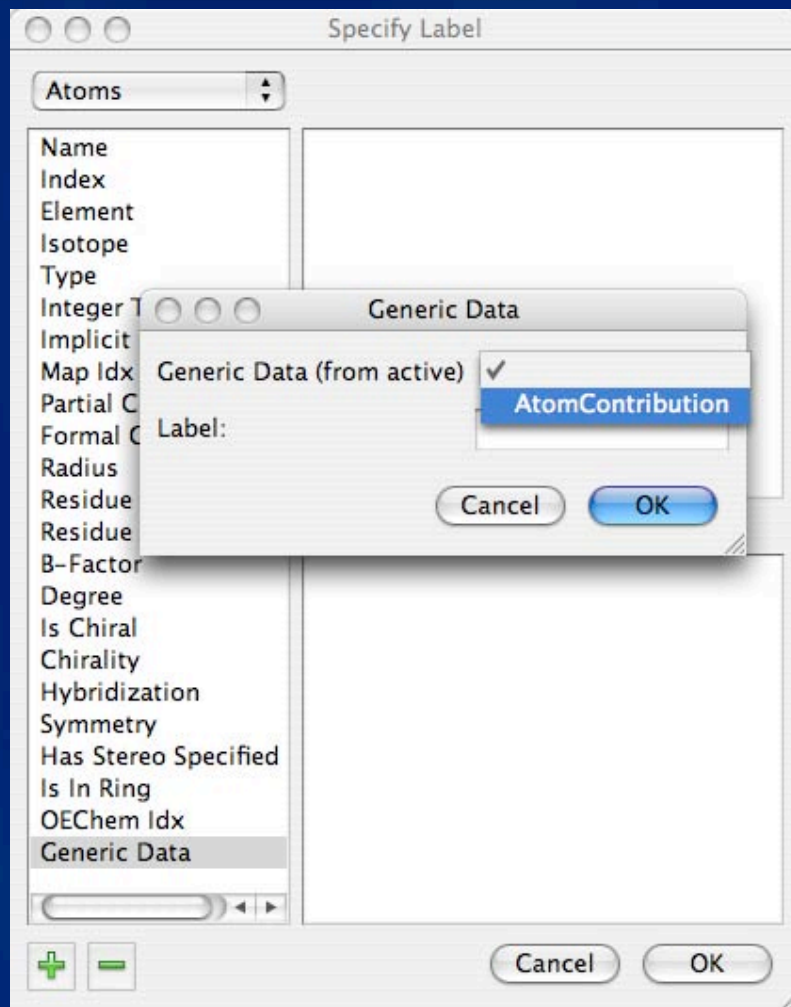
```
OEMultiplyScalarGrid(atmgrd, valuegrid)
```

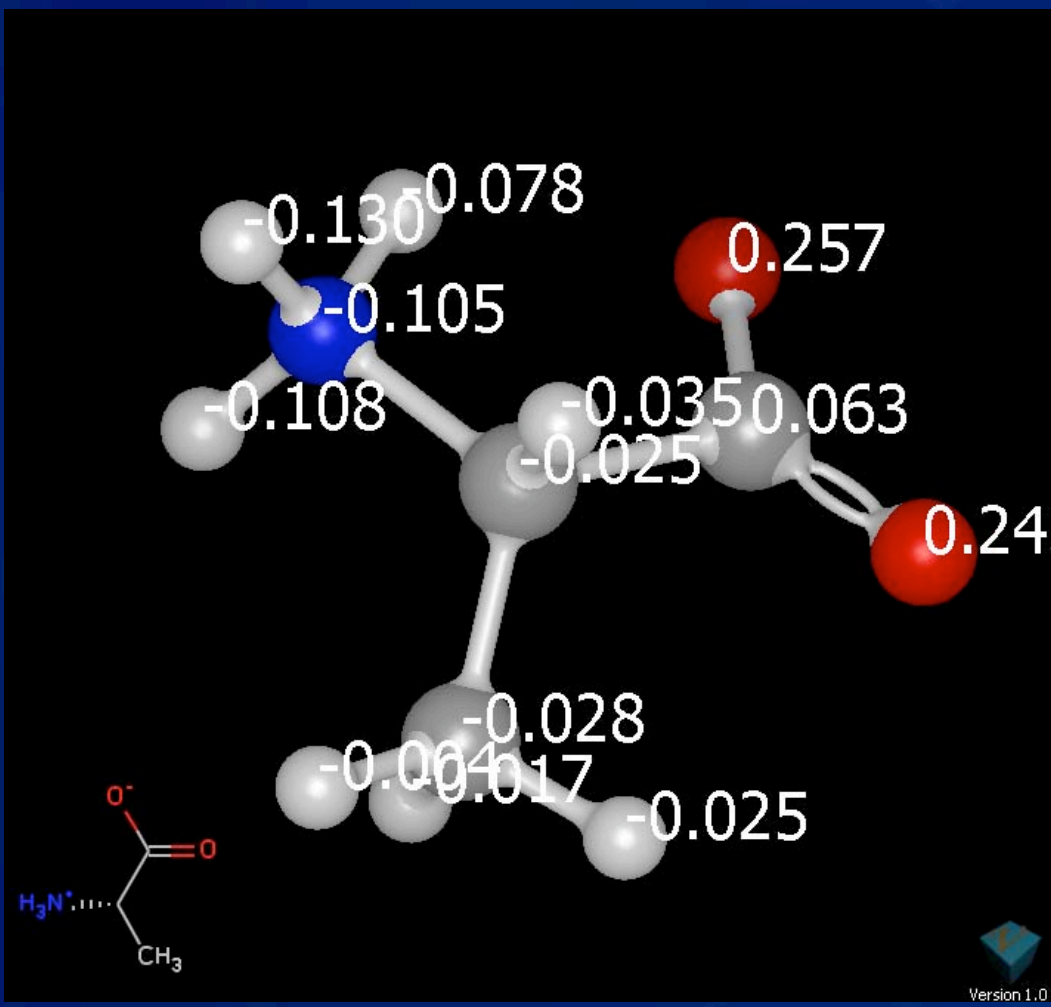
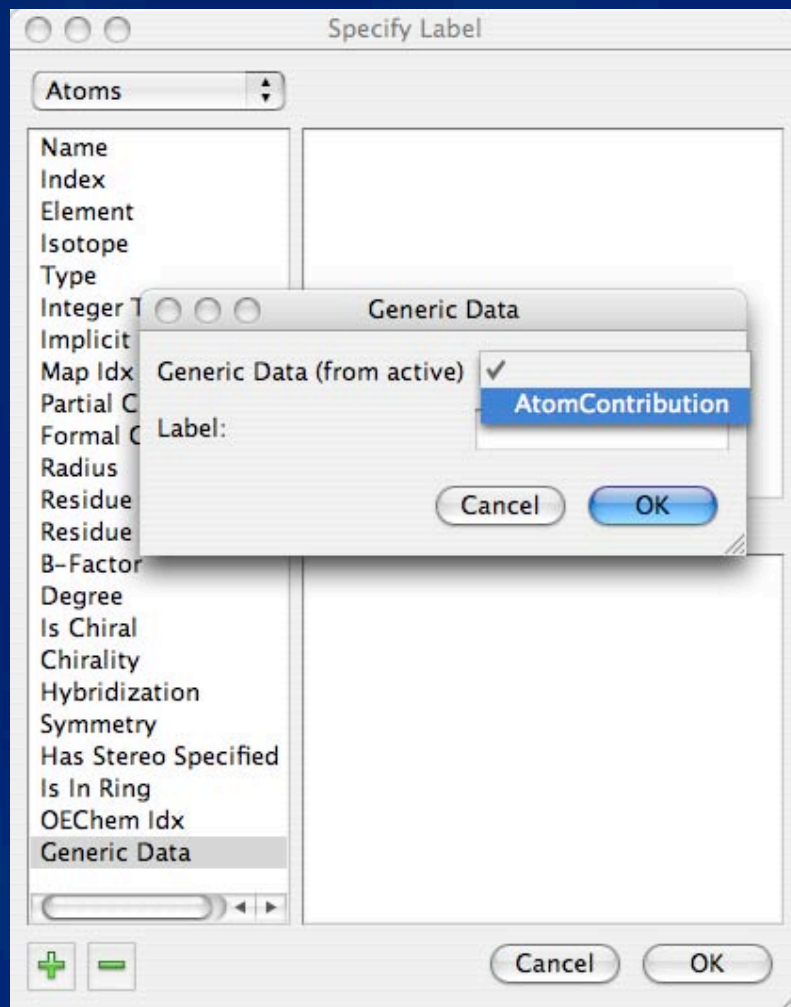
```
atomval = sum(atmgrd.GetValues()) * valuegrid.GetSpacing()**3
```

```
print "%2i: %f" % (atom.GetIdx(), atomval)
atom.SetData("AtomContribution", atomval)
```



# Viewing atom generic data in Vida





Specify Label

Atoms

- Name
- Index
- Element
- Isotope
- Type
- Integer T
- Implicit
- Map Idx
- Partial C
- Formal C
- Radius
- Residue
- Residue
- B-Factor
- Degree
- Is Chiral
- Chirality
- Hybridization
- Symmetry
- Has Stereo Specified
- Is In Ring
- OEChem Idx
- Generic Data

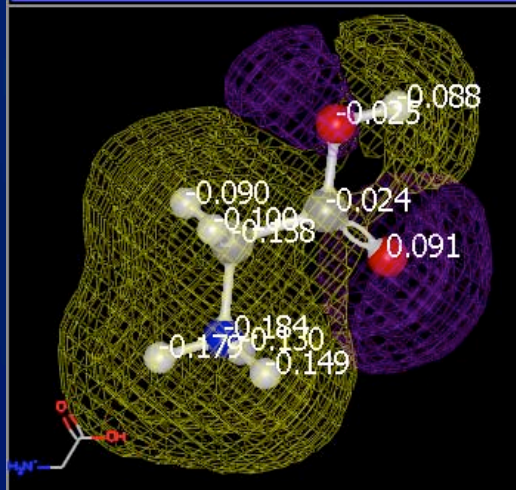
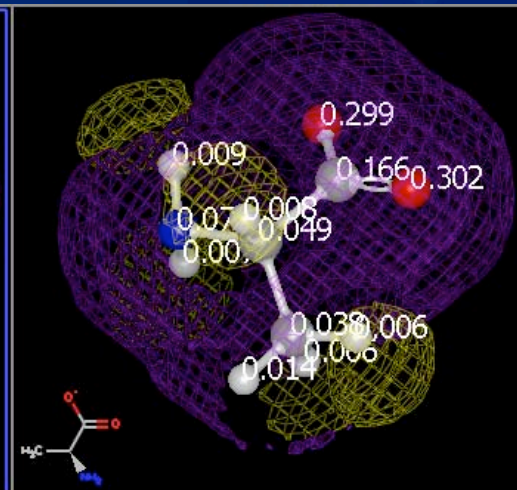
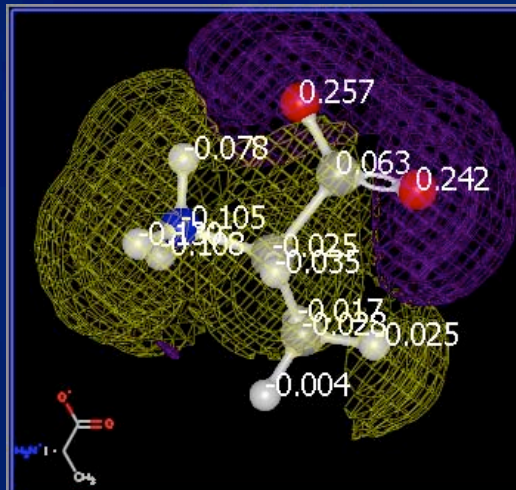
Generic Data

Generic Data (from active)

Label: AtomContribution

Cancel OK

Cancel OK



Specify Label

Atoms

- Name
- Index
- Element
- Isotope
- Type
- Integer T
- Implicit
- Map Idx
- Partial C
- Formal C
- Radius
- Residue
- Residue
- B-Factor
- Degree
- Is Chiral
- Chirality
- Hybridization
- Symmetry
- Has Stereo Specified
- Is In Ring
- OEChem Idx
- Generic Data

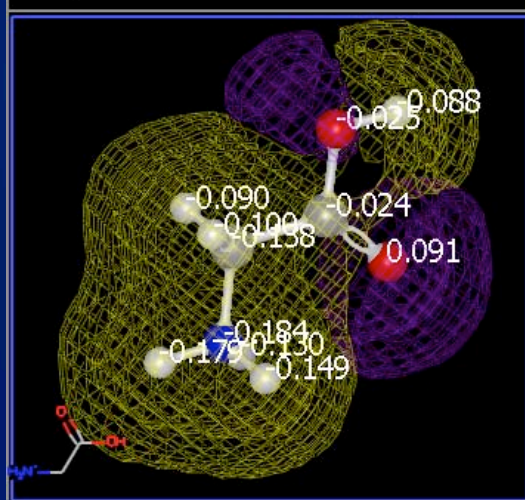
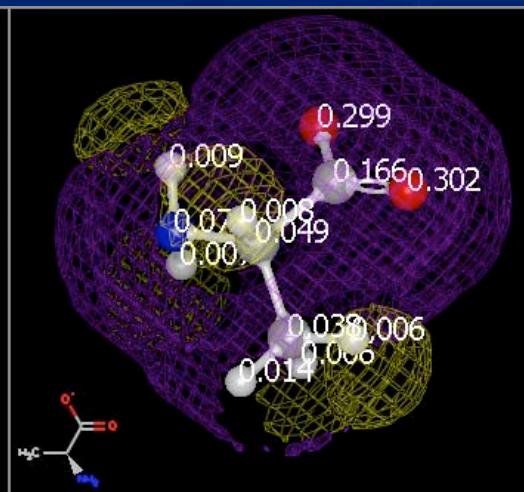
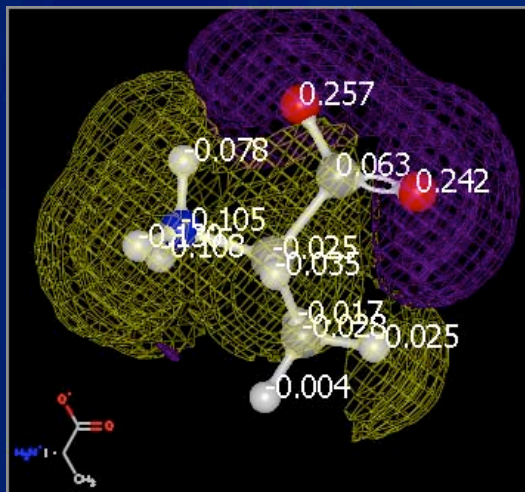
Generic Data

Generic Data (from active)

Label: **AtomContribution**

Cancel OK

Cancel OK



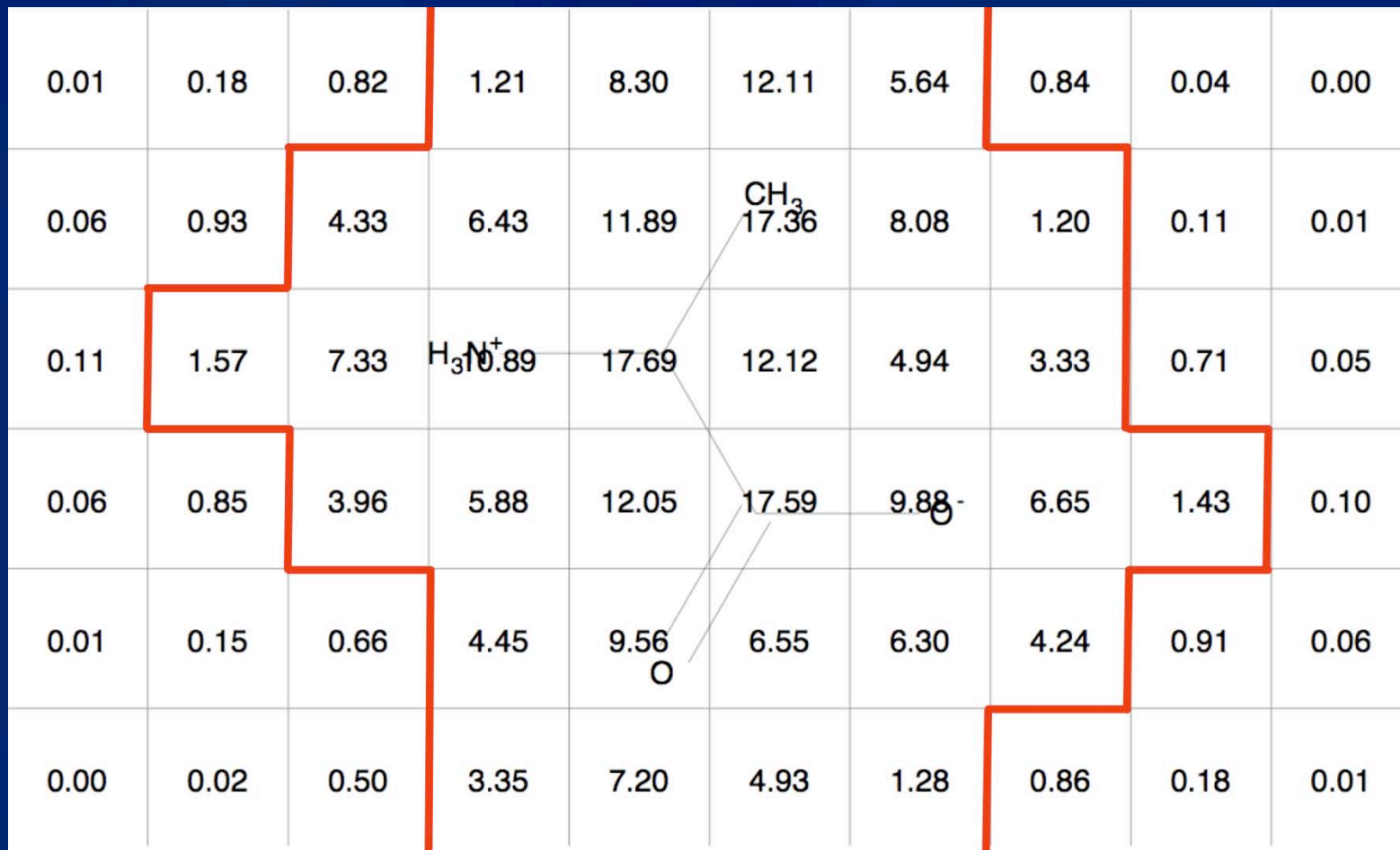
# Analyzing induced charge

- **Choosing contours**
  - Find maximum and minimum value in grid
  - Multiply each by 0.1
- **Analyzing the contours**
  - Make a surface from each of the contours above
  - Calculate the enclosed volume
  - Calculate the surface area



# Visualizing a mess of values

- Contours



# Choosing contours

```
def ContourGrid(grid, percentile):
```

```
    vals = grid.GetValues()
```

```
    high = max(vals)
```

```
    low = min(vals)
```

```
    highcontour = high * percentile
```

```
    lowcontour = low * percentile
```

```
    AddContour(grid, highcontour, "PositiveInducedCharge", OEPurple)
```

```
    AddContour(grid, lowcontour, "NegativeInducedCharge", OEYellow)
```



# OESpicoli

- Another core OpenEye library
- Supports surfaces as first class objects
  - OESurface
- Features
  - IO from several formats
  - Attach as generic data to molecules
  - Fast molecular and accessible surface construction
  - Cavity extraction
  - Surface <-> Grid interpolation

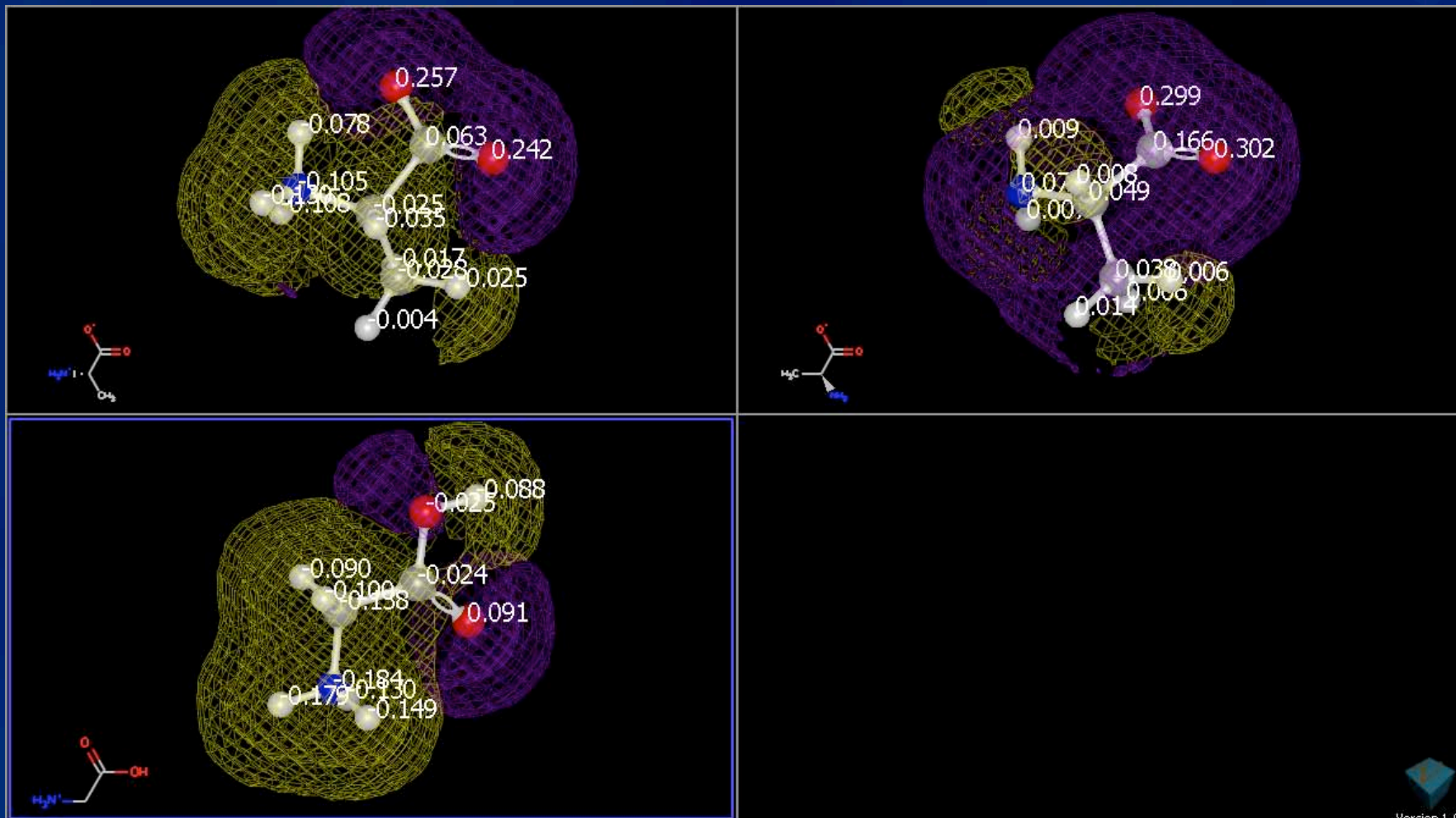


# Analyzing contours

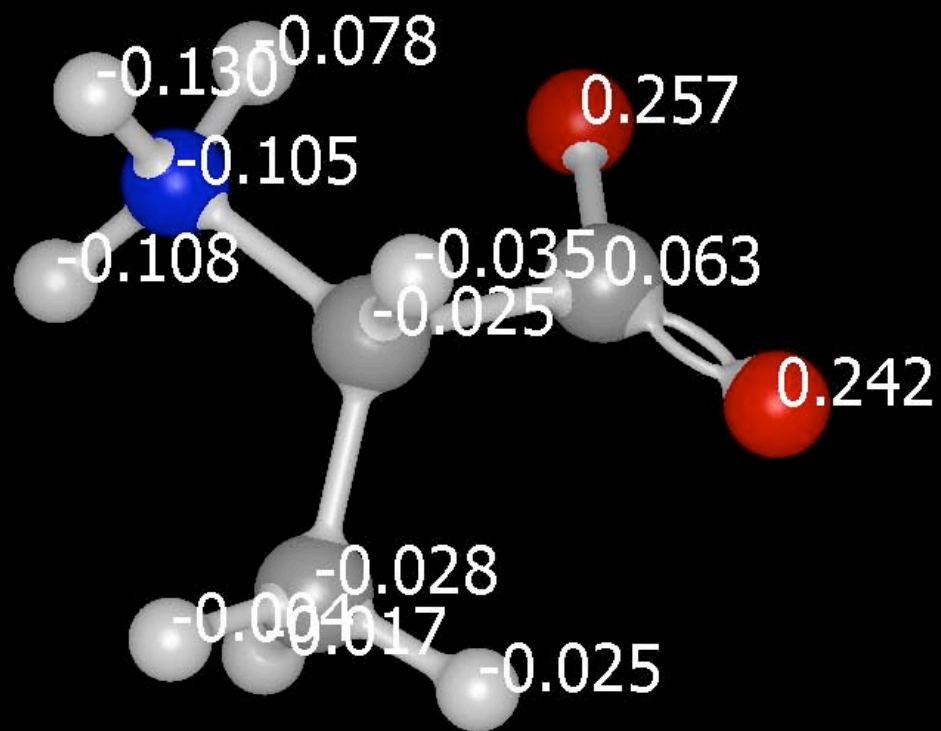
```
def AddContour(grid, contour, name, color):  
    OEAddContour(grid, contour, color)  
  
    surf = OESurface()  
    OEMakeSurfaceFromGrid(surf, grid, contour)  
  
    surf.SetTitle(name)  
    OESetSurfaceColor(surf, *color.GetRGBA())  
  
    surf.SetData("Volume", OESurfaceVolume(surf))  
    surf.SetData("Area", OESurfaceArea(surf))  
  
    grid.SetData(name, surf)
```



# View everything in Vida



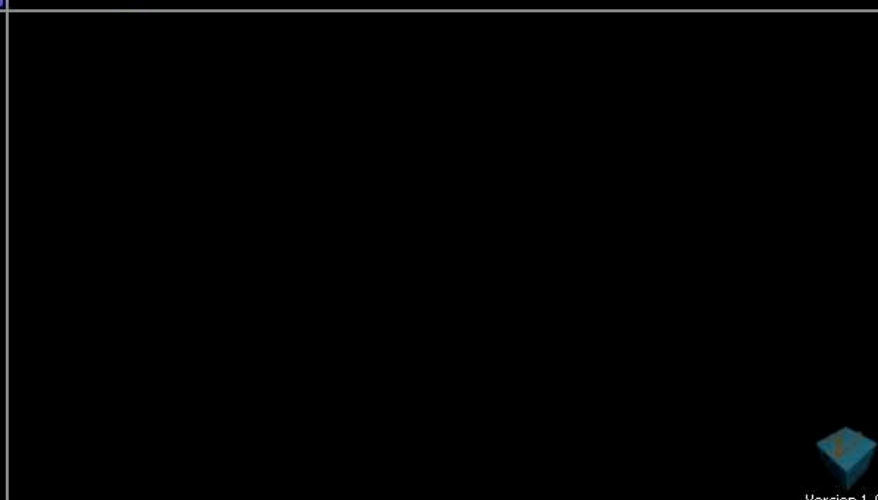
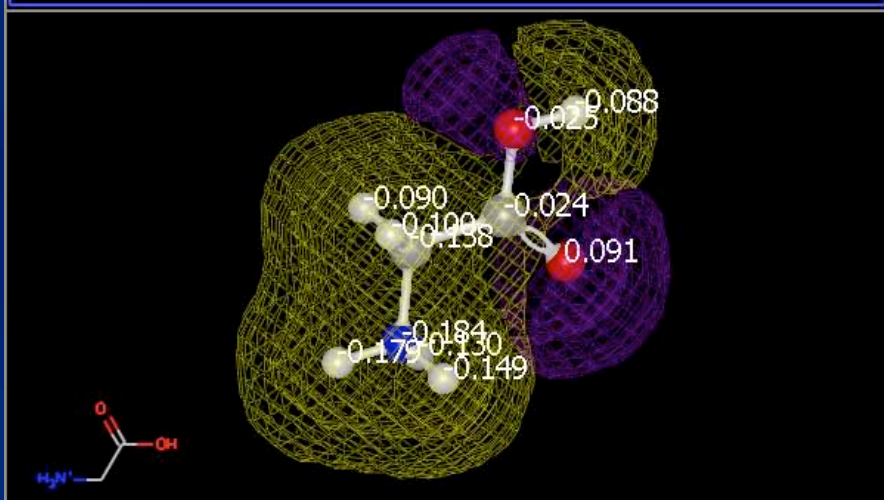
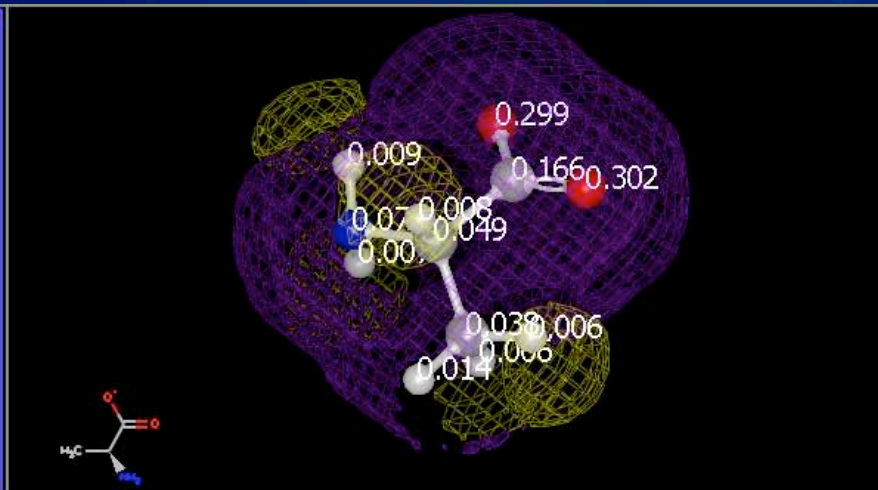
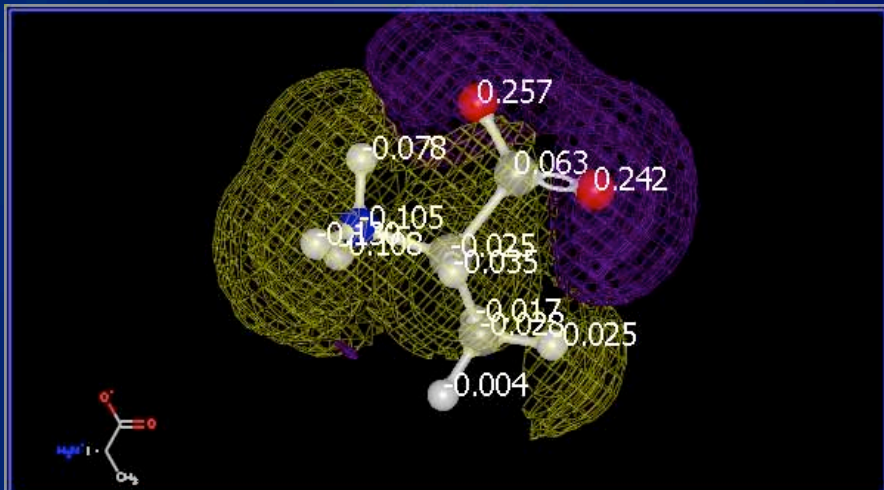
# View everything in Vida



Version 1.0

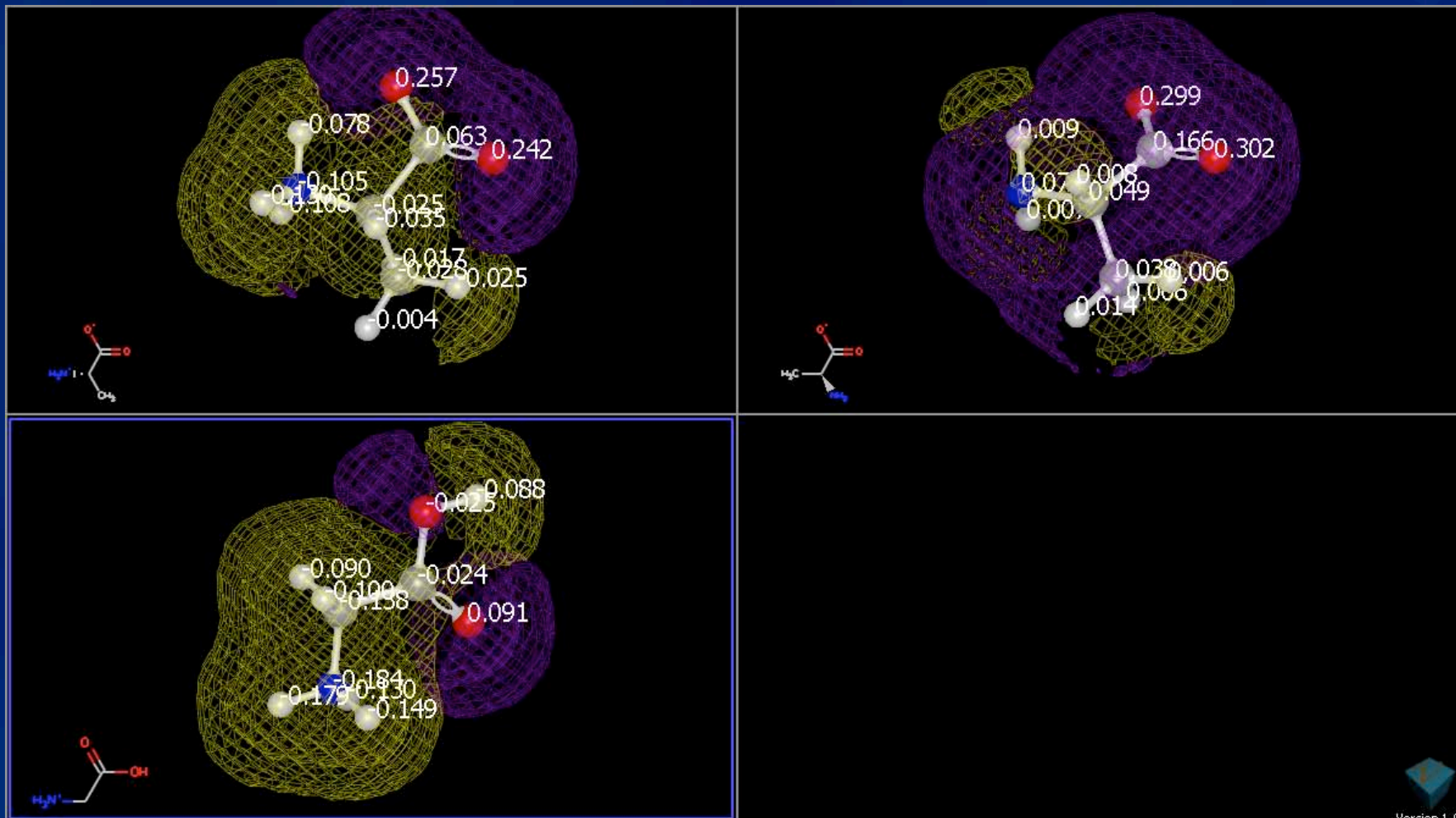


# View everything in Vida



Version 1.0

# View everything in Vida



Version 1.0