

Solving Gradient Equals Zero

When solving for when the gradient is equal to zero, Sage (and the underlying program Maxima) cannot always find an algebraic solution.

```
# We define the function and its derivatives.
x1,x2=var('x1,x2')
f=(1-x1+x1^2)*e^(x2^2) + (1-x2+x2^2)*e^(x1^2)
fd1=f.diff(x1)
fd2=f.diff(x2)
solve([fd1==0,fd2==0],x1,x2)
```

Traceback (click to the left for traceback)

```
...
ValueError: Unable to solve [(2*x1 - 1)*e^x2^2 + 2*x1*e^x1^2*(x2^2 -
x2 + 1) == 0, 2*(x1^2 - x1 + 1)*x2*e^x2^2 + e^x1^2*(2*x2 - 1) == 0]
for (x1, x2)
```

In this case, Sage cannot find an algebraic solution. Instead, we attempt to find a numeric solution.

We use the `fsolve` procedure from the SciPy package in Sage.

```
from scipy.optimize import fsolve # import the function so it is
available to use

def evalfunc(v): # fsolve only works with a Python function, not a
Sage mathematical function.
# The input v is a list [v[0],v[1]] of the parameters that should be
treated as a vector.

return
[fd1.substitute(x1=v[0],x2=v[1]),fd2.substitute(x1=v[0],x2=v[1])] #
evaluate the gradient

sol=fsolve(evalfunc,[float(1.6),float(7)]) # use as initial point
(x1,x2)=(1.6,7)

print sol
```

```
[ 0.27788009  0.27788009]
```

For different initial points, `fsolve` may produce different solutions. In this case, however, there is only one solution.

```
print f.substitute(x1=sol[0],x2=sol[1])
```

```
1.727011051424758
```