fit

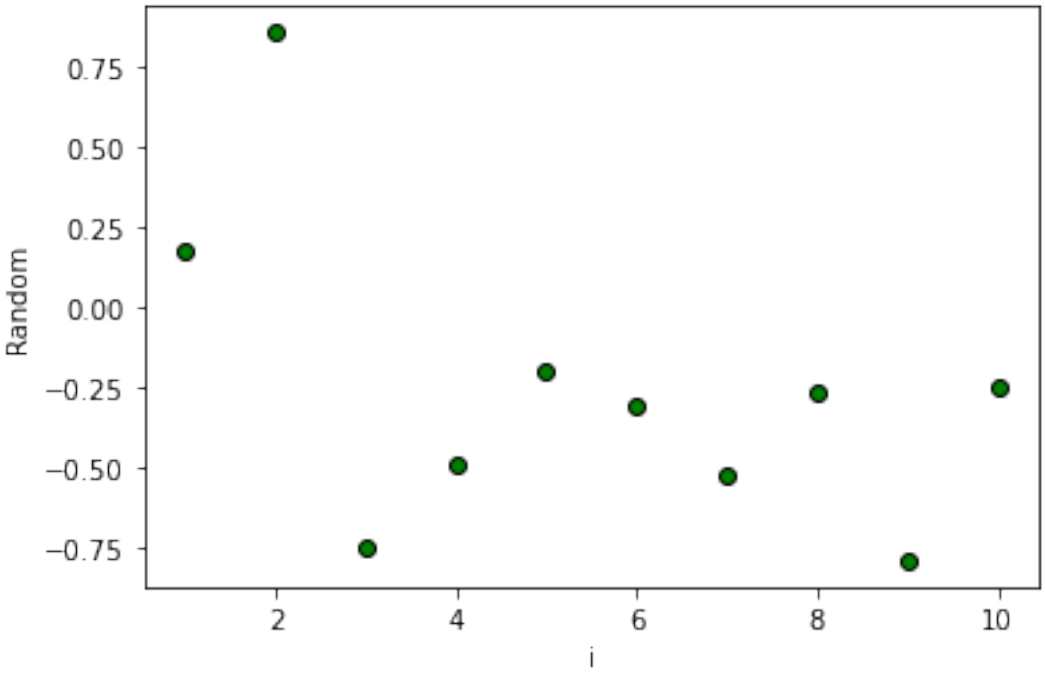
June 28, 2021

[1]:
```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LinearRegression
%matplotlib inline
```

0.1 Plotting the randomly generated points

[2]:
```
x_data = np.array(range(1,11))
y_data = -1+2*np.random.rand(10)
```

[3]:
```
plt.figure()
plt.scatter(x_data, y_data, c='green', edgecolors='black')
plt.xlabel("i")
plt.ylabel("Random")
plt.savefig("Points.png", dpi=300)
plt.show()
```



0.2 Fitting the Points

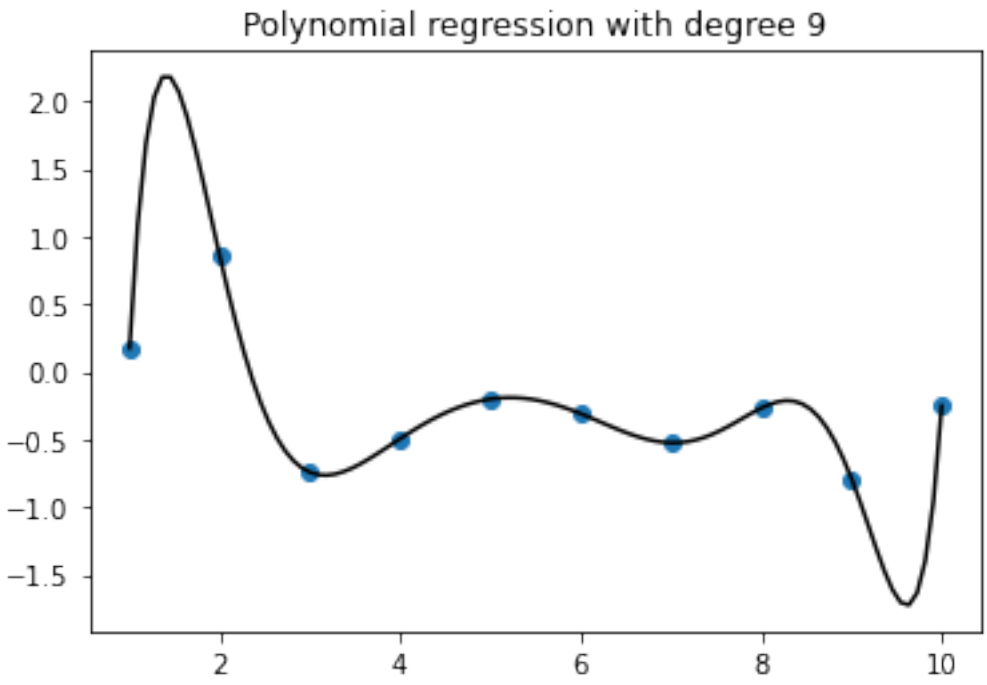
To fit n points perfectly, we require a polynomial of degree $n-1$

[4]:
```
# Using SciKit Learn to fit a nth degree polynomial
degree=9
polyreg=make_pipeline(PolynomialFeatures(degree),LinearRegression())
polyreg.fit(x_data.reshape(-1,1),y_data)
```

[4]:
```
Pipeline(steps=[('polynomialfeatures', PolynomialFeatures(degree=9)),
                ('linearregression', LinearRegression())])
```

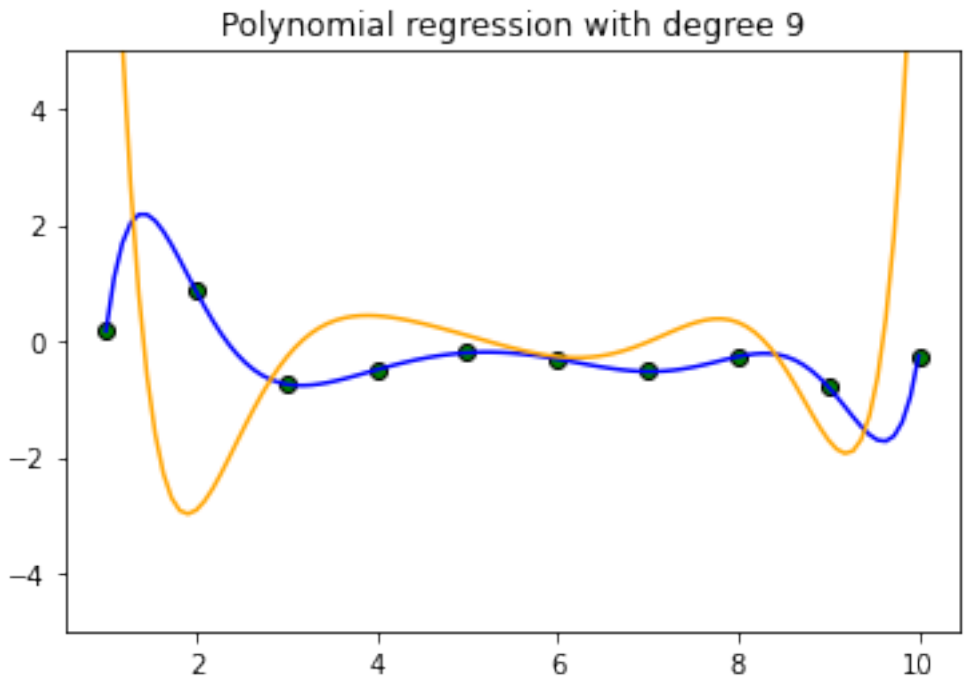
[5]:
```
# Sampling the fitted polynomial
x_fit = np.linspace(1,10, num=100)
y_fit = polyreg.predict(x_fit.reshape(-1,1))
```

[6]:
```
plt.figure()
plt.scatter(x_data,y_data)
plt.plot(x_fit, y_fit,color="black")
plt.title("Polynomial regression with degree "+str(degree))
plt.savefig("Fitted.png", dpi=300)
plt.show()
```



[7]:
```
# Computing the gradient
grad = np.gradient(y_fit, x_fit)
```

[8]:
```
plt.figure()
plt.scatter(x_data, y_data, c='green', edgecolors='black')
plt.plot(x_fit, y_fit,color="blue")
plt.plot(x_fit, grad, color="orange")
plt.ylim([-5,5])
plt.title("Polynomial regression with degree "+str(degree))
plt.show()
```



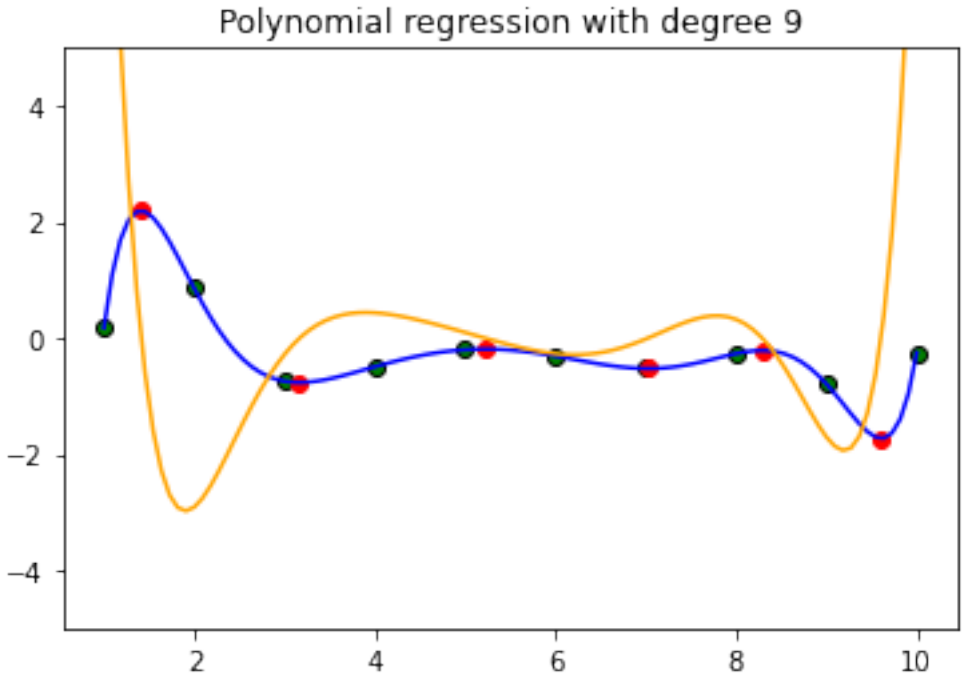
0.2.1 Finding the extremas

[9]:
```
x_find = np.linspace(0.1,9.9, num=10000)
extrema_x = []
extrema_y = []

for t in range(1, len(x_find)-1):
    data = x_find[t-1:t+2].reshape(-1,1)
    fit = polyreg.predict(data)

    if fit[1]>fit[0] and fit[1]>fit[2]:
        print("Maxima at x =", x_find[t], "and y =", fit[1])
        extrema_x.append(x_find[t])
        extrema_y.append(fit[1])
    elif fit[1]<fit[0] and fit[1]<fit[2]:
        print("Minima at x =", x_find[t], "and y =", fit[1])
        extrema_x.append(x_find[t])
        extrema_y.append(fit[1])
```

```
Maxima at x = 1.4084308430843084 and y = 2.195104229252479
Minima at x = 3.164766476647665 and y = -0.7656559740882756
Maxima at x = 5.218071807180718 and y = -0.19116876616108414
Minima at x = 7.025372537253725 and y = -0.5251533421000687
Maxima at x = 8.292639263926393 and y = -0.2132152812141186
Minima at x = 9.61087108710871 and y = -1.7288452082816121
```

Checking the results obtained

[10]:
```
plt.figure()
plt.scatter(x_data, y_data, c='green', edgecolors='black')
plt.scatter(extrema_x, extrema_y, c='red')
plt.plot(x_fit, y_fit,color="blue")
plt.plot(x_fit, grad, color="orange")
plt.ylim([-5,5])
plt.title("Polynomial regression with degree "+str(degree))
plt.show()
```



[ ]: