RANSAC

One way of dealing with outliers ...

UNIVERSITY OF BIRMINGHAM

```
#!/usr/bin/env python3

import numpy as np
import matplotlib.pyplot as plt
#import
from sklearn import linear_model, metrics
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.externals import joblib
import seaborn as sns
```

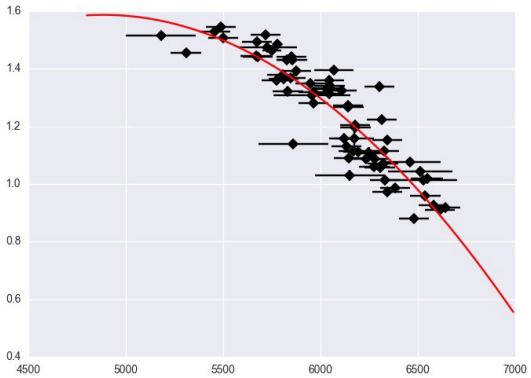
```
#!/usr/bin/env python3

import numpy as np
import matplotlib.pyplot as plt
#import
from sklearn import linear_model, metrics
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.externals import joblib
import seaborn as sns
```

```
def get_data(sfile="Data/LEGACY_teff_ep.txt"):
    k, t, te, ep, epe = np.genfromtxt(sfile).T
    n_extra = 30
    k = np.append(k, np.random.randn(n_extra))
    t = np.append(t, np.random.randn(n_extra)*800 + 5500.0)
    te = np.append(te, np.random.randn(n_extra)*80.0)
    ep = np.append(ep, np.random.randn(n_extra)*1.0)
    epe = np.append(epe, np.random.randn(n_extra)*0.05)
    return k, t, te, ep, epe
```

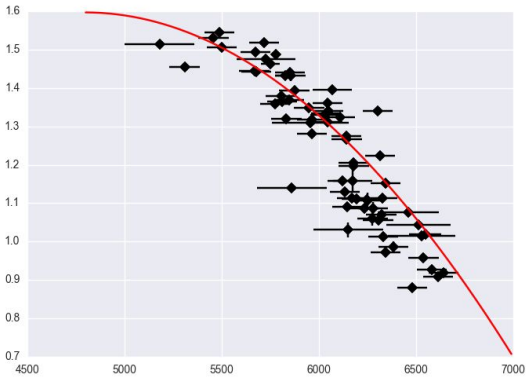
```
# use 3 different estimators to fit the Teff/epsilon data
def est(t, te, ep, epe):
    xt = np.arange(4800, 7000, 10)
    estimators = [('OLS', linear_model.LinearRegression()),
                  ('Theil-Sen', linear_model.TheilSenRegressor(random_state=42)),
                  ('RANSAC', linear_model.RANSACRegressor(random_state=42)), ]
    for name, estimator in estimators:
        model = make_pipeline(PolynomialFeatures(2), estimator)
        model.fit(t[:, np.newaxis], ep)
        y_plot = model.predict(xt[:, np.newaxis])
        # get the coefficients of the RANSAC fit to the data and put them
        # in a pickle
        if name == 'RANSAC':
            print(estimator.estimator_.coef_)
            coef = estimator.estimator_.coef_.flatten()
            joblib.dump(model, 'teff_ep_RANSAC.pkl')
            inlier_mask = estimator.inlier_mask_
            plot_(t, te, ep, epe, model=y_plot, modelx=xt, show=True, im=inlier_mask)
        else:
            print(estimator.coef_)
            plot_(t, te, ep, epe, model=y_plot, modelx=xt, show=True)
        print("Estimator : ", name)
    plt.show()
```

Fit a model to some data ...

Standard fit



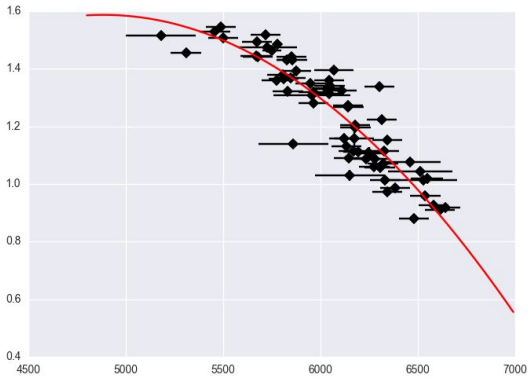
Theil-Sen



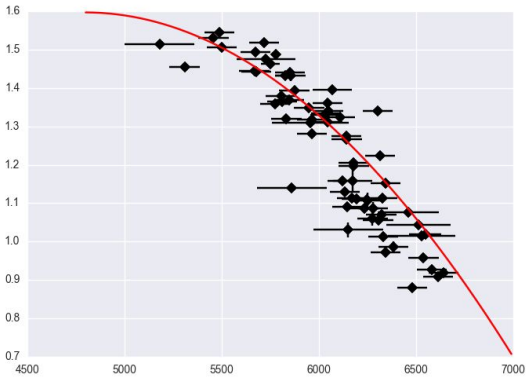
UNIVERSITY OF BIRMINGHAM

Fit a model to some data ...

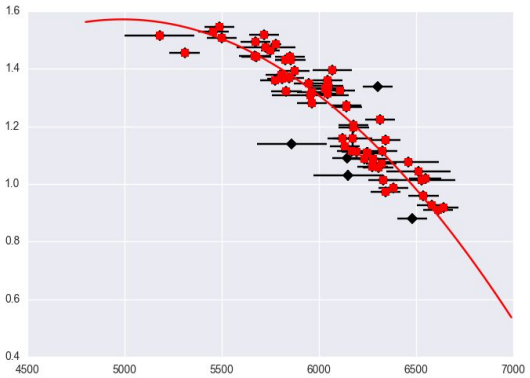
Standard fit



Theil-Sen



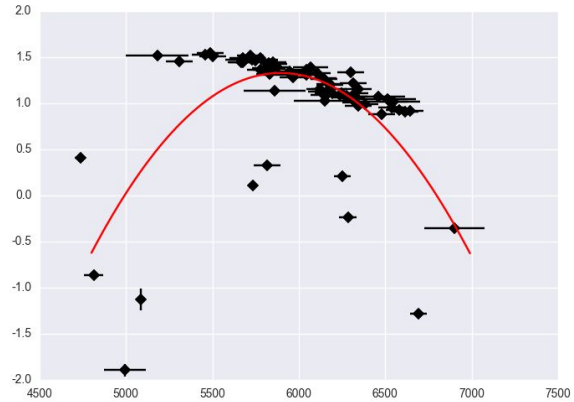
RANSAC



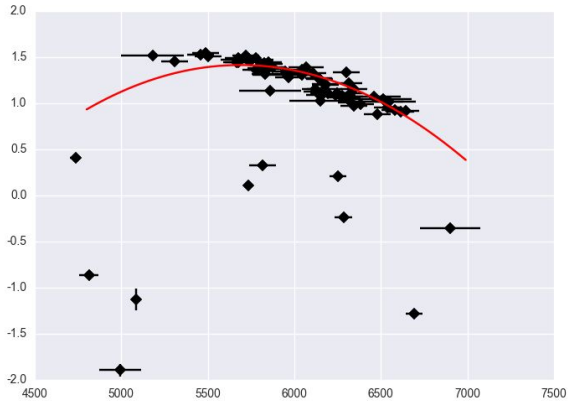
UNIVERSITY OF BIRMINGHAM

Fit a model to some data ...

Standard fit



Theil-Sen

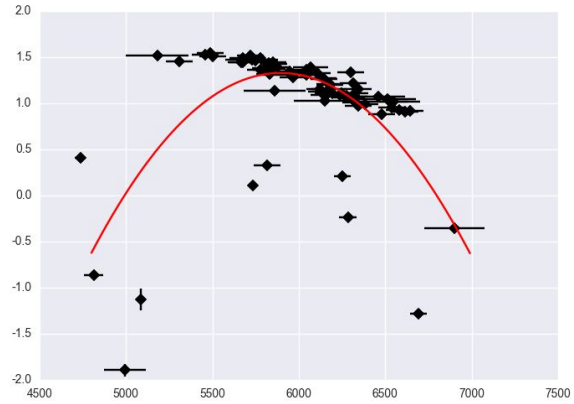


RANSAC

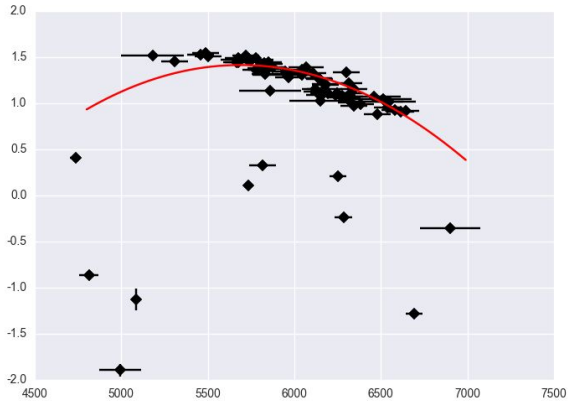
UNIVERSITY OF BIRMINGHAM

Fit a model to some data ...

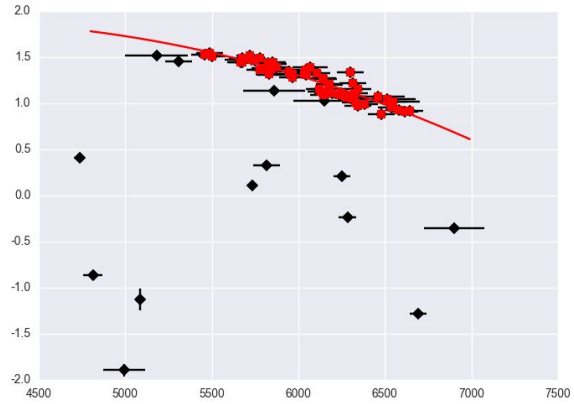
Standard fit



Theil-Sen



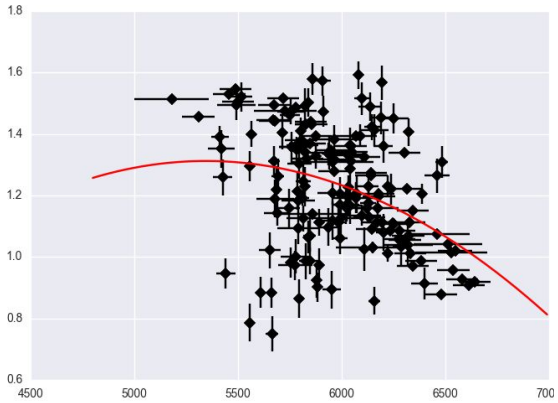
RANSAC



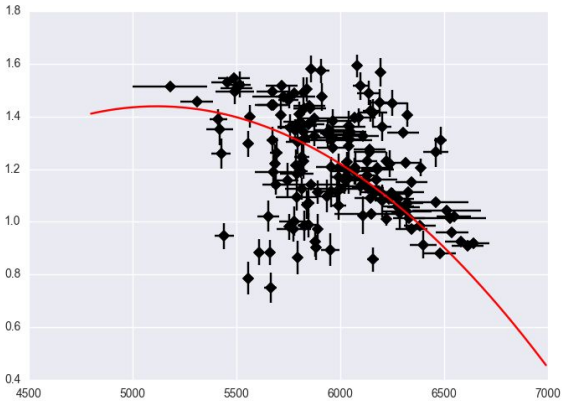
UNIVERSITY OF BIRMINGHAM

Fit a model to some data ...

Standard fit



Theil-Sen

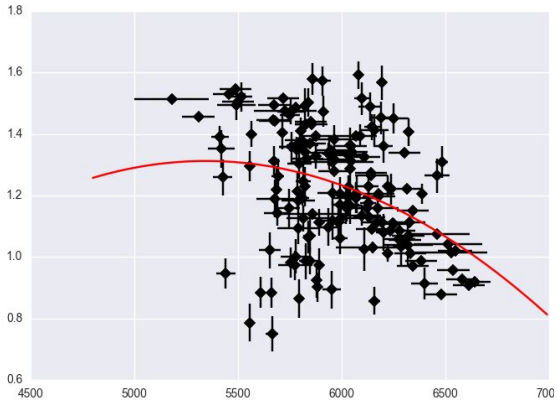


RANSAC

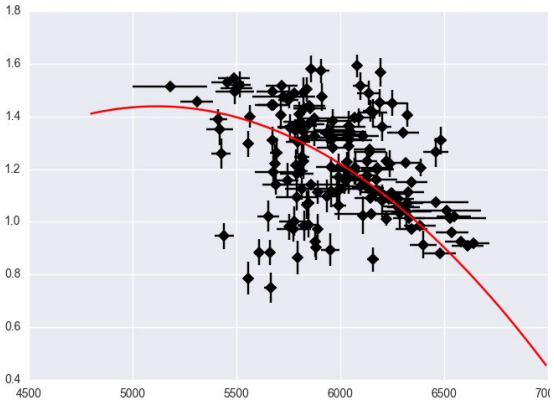
UNIVERSITY OF BIRMINGHAM

Fit a model to some data ...

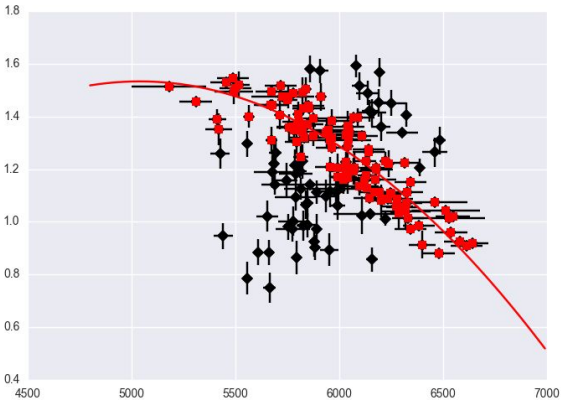
Standard fit



Theil-Sen

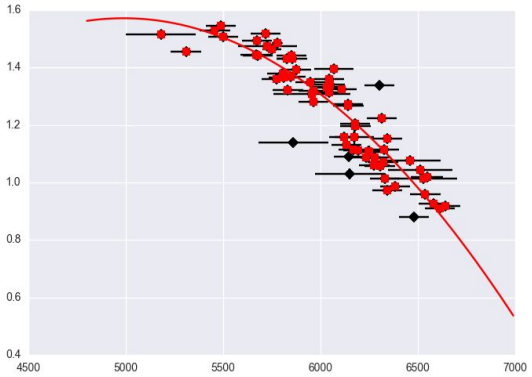


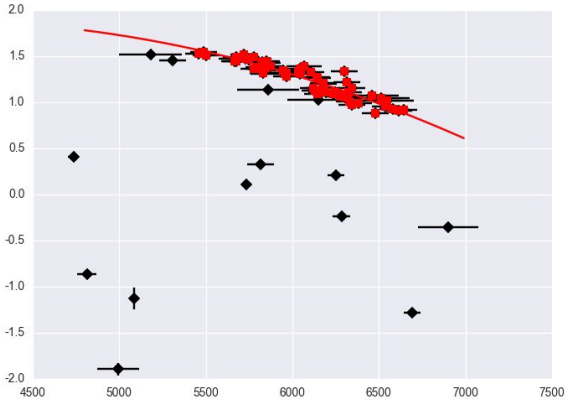
RANSAC

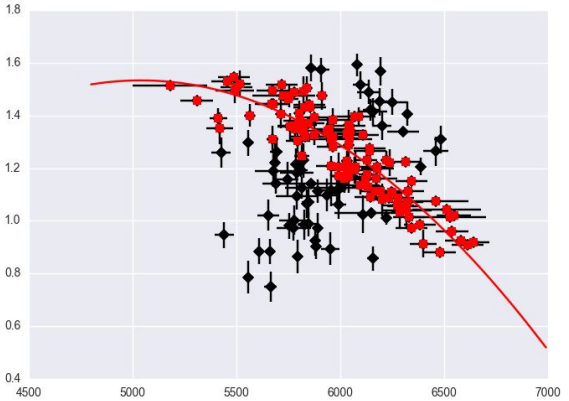


UNIVERSITY OF BIRMINGHAM

Fit a model to some data ...







UNIVERSITY OF BIRMINGHAM