HW7 S23 - Starter Code

Q1: NFL Regression

In [19]:
```
using DataFrames, DelimitedFiles
using PyPlot, JuMP, HiGHS, Ipopt
```

In [20]:
```
teams = [ "49ers", "falcons", "ravens", "steelers", "bears", "vikings", "lions",
teamIDs = Dict( zip( teams, Array(1:31) ) )
data = readdlm( joinpath(@__DIR__, "nfl.inc" ), header=false)
results_table=Array{Int16,2}(undef, size(data,1),4)
for i in 1:size( data, 1 )
    base = data[ i, 1 ]
    res = data[ i, 2 ]
    splits = split( base, "." )
    week = parse( Int64, splits[1] )
    away = splits[2]
    home = splits[3]
    aid = teamIDs[away]
    hid = teamIDs[home]
    results_table[i,1]=week
    results_table[i,2]=aid
    results_table[i,3]=hid
    results_table[i,4]=res
end
# print(results_table)


n_terms=size(results_table,1)
n_teams=31
```

Out[20]:
```
31
```

In [21]:
```
# Solution for (a)

# your code here

m = Model(HiGHS.Optimizer)

@variable(m, ratings[1:n_teams])
@variable(m, home_field)
@constraint(m, sum(ratings[i] for i in 1:n_teams) == 0)

@expression(m, pred[i in 1:n_terms], ratings[results_table[i, 3]]-ratings[result
@objective(m, Min, sum((pred[i] - results_table[i, 4])^2 for i in 1:n_terms))

optimize!(m)
ratingsa = value.(ratings)
hfa = value(home_field)

# print results
for i=1:n_teams
    println(teams[i]," : ",value(ratings[i]))
```

```
end
    println("Home field advantage: ", value(home_field))
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
0, 29333.000000, 31, 0.000000, 0.000000, 0, 0.000000, 0.000000
3, 17134.840804, 31, 0.000000, 0.000000, 0, 0.000000, 1.000000
Model   status      : Optimal
QP ASM    iterations: 3
Objective value     :  1.7134840737e+04
HiGHS run time      :          0.00
49ers : -7.4447953373501505
falcons : -8.16224547202938
ravens : 4.840011659429398
steelers : 3.1099535124758995
bears : -6.193788456506018
vikings : 0.9390918876671944
lions : 2.354697530270662
saints : 0.487070791469117
bucs : 9.242807323623529
patriots : 2.423993496977115
jets : 4.770665202392323
packers : -0.6581268257660479
chargers : -6.289431917134701
raiders : 7.964138099652889
cardinals : -14.364741602678672
giants : 2.438494881805434
panthers : 0.2763420319238483
redskins : 1.8445897427877407
jaguars : -5.741125251530226
browns : -15.102694387588244
colts : 7.8157988686033875
chiefs : 6.581879367904487
eagles : 2.6114502616705457
cowboys : -4.3576225968048705
seahawks : -1.3183229158451977
dolphins : 4.386500259923016
titans : 8.145583282397332
bills : 1.932251815064968
broncos : 5.056835190978192
rams : 5.130971873029371
bengals : -12.72023231681294
Home field advantage: 1.9144730252546178
```

comment:

The differences in the ratings between these two methods are their values are not exactly the same, though they have similar trend. The ratings (predicted outcomes) with sum-of-squares is more closer to the actual outcomes.

In [13]:
```
# Solution for (b)

# your code here

m = Model(HiGHS.Optimizer)

@variable(m, ratings[1:n_teams])
@variable(m, t[1:n_terms])
@variable(m, home_field)
```

```
@constraint(m, sum(ratings[i] for i in 1:n_teams) == 0)

@expression(m, pred[i in 1:n_terms], ratings[results_table[i, 3]]-ratings[result
for i in 1:n_terms
    @constraint(m, -t[i] <= pred[i] - results_table[i, 4])
    @constraint(m, pred[i] - results_table[i, 4] <= t[i])
end
@objective(m, Min, sum(t[i] for i in 1:n_terms))

optimize!(m)
ratingsb = value.(ratings)
hfb = value(home_field)

for i=1:n_teams
    println(teams[i]," : ",value(ratings[i]))
end
println("Home field advantage: ", value(home_field))
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
Presolving model
287 rows, 175 cols, 1175 nonzeros
287 rows, 175 cols, 1175 nonzeros
Presolve : Reductions: rows 287(-0); columns 175(-0); elements 1175(-0) - Not r
educed
Problem not reduced by presolve: solving the LP
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0    -1.4300000000e+05 Ph1: 144(317000); Du: 143(143) 0s
        305     1.1745714286e+03 Pr: 0(0) 0s
Model   status      : Optimal
Simplex   iterations: 305
Objective value     :  1.1745714286e+03
HiGHS run time      :          0.01
49ers : -7.073732718894009
falcons : -6.073732718894009
ravens : -0.6451612903225805
steelers : 3.497695852534562
bears : -3.7880184331797224
vikings : 0.35483870967741993
lions : 4.35483870967742
saints : -0.7880184331797242
bucs : 6.069124423963132
patriots : -0.07373271889401067
jets : 4.0691244239631335
packers : -1.0737327188940098
chargers : -2.930875576036867
raiders : 5.069124423963135
cardinals : -12.930875576036867
giants : 2.0691244239631335
panthers : -1.359447004608295
redskins : 0.4976958525345623
jaguars : -6.788018433179722
browns : -19.788018433179722
colts : 7.211981566820274
chiefs : 5.783410138248847
eagles : -0.9308755760368652
cowboys : -2.7880184331797224
seahawks : -0.07373271889401067
dolphins : 5.783410138248847
titans : 7.640552995391705
bills : 4.069124423963132
broncos : 12.211981566820276
rams : 8.783410138248849
bengals : -10.359447004608295
Home field advantage: 1.1428571428571428
```

In [15]:

```julia
# Solution for (c)

# your code here

m = Model(HiGHS.Optimizer)

@variable(m, ratings[1:n_teams])
@variable(m, home_field[1:n_teams])
@constraint(m, sum(ratings[i] for i in 1:n_teams) == 0)

@expression(m, pred[i in 1:n_terms], ratings[results_table[i, 3]]-ratings[result
```

```
@objective(m, Min, sum((pred[i] - results_table[i, 4])^2 for i in 1:n_terms))

optimize!(m)

# print results:
for i=1:n_teams
    println(teams[i]," : ",value(ratings[i]),"   home field advantage: ", value(
end
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
0, 29333.000000, 61, 0.000000, 0.000000, 0, 0.000000, 0.000000
3, 13995.507564, 61, 0.000000, 0.000000, 0, 0.000000, 0.948704
Model   status      : Optimal
QP ASM    iterations: 3
Objective value     :  1.3995507351e+04
HiGHS run time      :          0.00
49ers : -6.640258153096028   home field advantage: -2.8136208323501988
falcons : -11.99643159634961   home field advantage: 8.16298557145395
ravens : 4.884164738313829   home field advantage: -0.4056746536817149
steelers : 5.542330180118024   home field advantage: -7.004297051004981
bears : -5.032693481639377   home field advantage: 3.1883166003828496
vikings : -3.2499911433372612   home field advantage: 7.991445632878648
lions : 13.161143407657567   home field advantage: -15.165670267176484
saints : 2.0868085304771355   home field advantage: 5.097232265047393
bucs : 5.417221622167478   home field advantage: 11.491344772388363
patriots : 10.967413337492966   home field advantage: -11.92252033576743
jets : 11.755807370109686   home field advantage: -9.392782945089687
packers : 3.3087804382391712   home field advantage: -1.663292372503217
chargers : -5.562782548377882   home field advantage: 0.42311569694428575
raiders : 7.535907400792645   home field advantage: 1.5271151858119583
cardinals : -23.757724198528408   home field advantage: 15.72036974947397
giants : 6.729251543956879   home field advantage: -6.527924384674569
panthers : -0.17929747107994243   home field advantage: 1.2932246826204898
redskins : 2.170586959025697   home field advantage: -2.03285375874684
jaguars : -3.0790846760826485   home field advantage: -3.8011428818845028
browns : -15.182747386285849   home field advantage: 2.574481257925596
colts : 7.050903414378867   home field advantage: 10.345441323768249
chiefs : 5.242026259890901   home field advantage: 4.098804742452021
eagles : 6.394575653478132   home field advantage: -4.820858940919306
cowboys : -1.0021301032424041   home field advantage: -8.018492640083677
seahawks : -8.905471236488545   home field advantage: 13.194357944993165
dolphins : -0.6692329319731841   home field advantage: 14.521953166255987
titans : 4.568707113592633   home field advantage: 9.78992359100412
bills : -0.7494768508171095   home field advantage: 8.173759926668009
broncos : 3.724087920640156   home field advantage: 4.53347717637848
rams : 3.5855981863497695   home field advantage: 4.48346539916291
bengals : -18.11799229938329   home field advantage: 7.382988116410922
```

solution (d) If the Dolphins were to visit the Seahawks, the expected winner according to the sum-of-squares model would be Dolphins with margin 3.7903501505135955. Using the l_1 model, the expected winner would be Dolphins with margin 4.714285714285715.

In [8]:
```
findall(x->x=="dolphins", teams) # 26
findall(x->x=="seahawks", teams) # 25

println(ratingsa[25]-ratingsa[26]+hfa)
println(ratingsb[25]-ratingsb[26]+hfb)
```

```
-3.7903501505135955
-4.714285714285715
```

Q2: Huber Loss

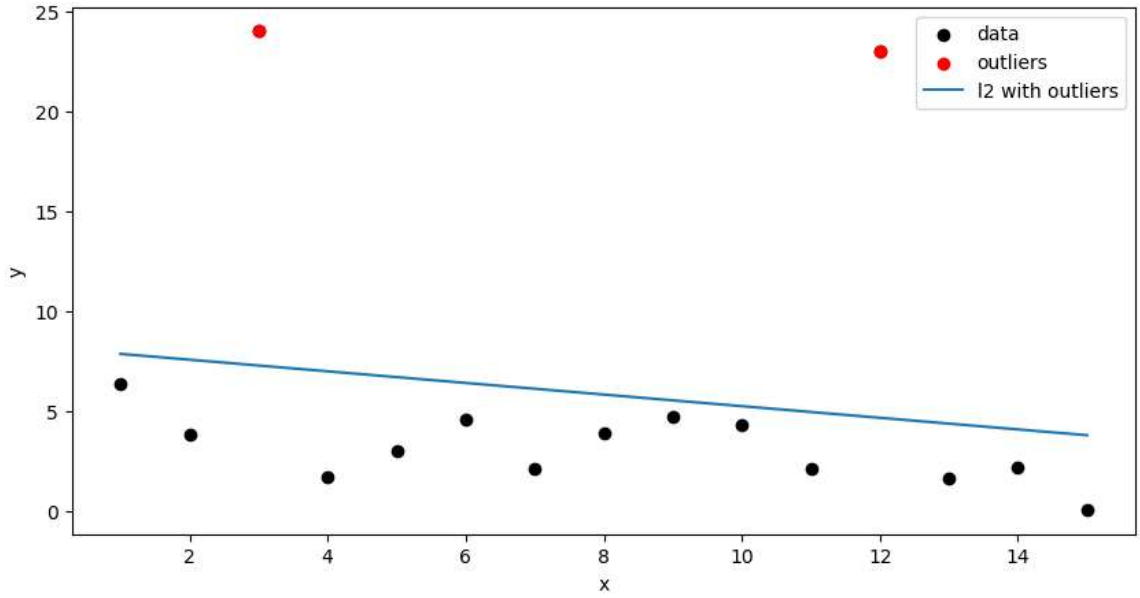
In [23]:
```
# Q2 Starter Code

using PyPlot, JuMP, Ipopt


y = [6.31, 3.78, 24, 1.71, 2.99, 4.53, 2.11, 3.88, 4.67, 4.25, 2.06, 23, 1.58, 2
x = 1:15;
```

a)

In [24]:
```
m = Model(HiGHS.Optimizer)

@variable(m, a1)
@variable(m, b1)

@expression(m, pred[i in x], a1 * i + b1)
@objective(m, Min, sum((pred[i] - y[i])^2 for i in x))

optimize!(m)
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
0, 1262.316800, 2, 0.000000, 0.000000, 0, 0.000000, 0.000000
3, 733.344790, 2, 0.000000, 0.000000, 0, 0.000000, 1.000000
Model   status      : Optimal
QP ASM    iterations: 3
Objective value     :  7.3334478714e+02
HiGHS run time      :          0.00
```

In [25]:
```
a1 = value(a1)
b1 = value(b1)

using PyPlot
figure(figsize = (10,5))
scatter(x,y,label="data",color = "black")
scatter([x[3],x[12]],[y[3],y[12]],label="outliers",color = "red") # outliers
plot(x, a1*x .+ b1,label="l2 with outliers")
# plot(x, a3*x .+ b3,label="l1 with outliers")
# plot(x, a5*x .+ b5,label="Huber with outliers")
legend(loc ="best")
ylabel("y")
xlabel("x")
;
```



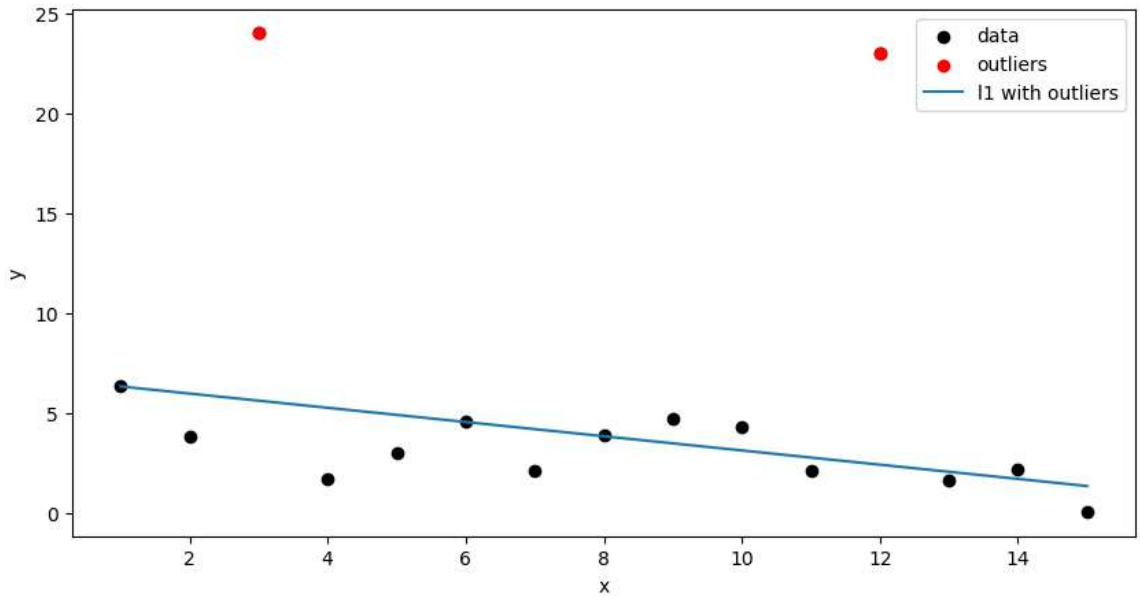
b)

The l1 cost handle outliers better than the least squares. It is more fitted to the good data points, because if lines are same, the cost of outliers with l1 norm is smaller than the cost of outliers with least squares. To minimize the cost, the line with least squares will be farther from non-outliers, nearer to outliers, compared to the line with l1 cost. So the line of l1 cost better fit the non-outliers.

In [26]:
```
m = Model(HiGHS.Optimizer)

@variable(m, a3)
@variable(m, b3)
@variable(m, t[x] >= 0)

@expression(m, pred[i in x], a3 * i + b3)
@constraint(m, -t .<= pred - y)
@constraint(m, pred - y .<= t)

@objective(m, Min, sum(t[i] for i in x))

optimize!(m)
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
Presolving model
30 rows, 17 cols, 90 nonzeros
30 rows, 17 cols, 90 nonzeros
Presolve : Reductions: rows 30(-0); columns 17(-0); elements 90(-0) - Not reduc
ed
Problem not reduced by presolve: solving the LP
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0     0.0000000000e+00 Pr: 15(72.555); Du: 0(1.12313e-12) 0s
         18     5.4030000000e+01 Pr: 0(0) 0s
Model   status      : Optimal
Simplex   iterations: 18
Objective value     :  5.4030000000e+01
HiGHS run time      :          0.01
```

In [27]:
```
a3 = value(a3)
b3 = value(b3)

using PyPlot
figure(figsize = (10,5))
scatter(x,y,label="data",color = "black")
scatter([x[3],x[12]],[y[3],y[12]],label="outliers",color = "red") # outliers
# plot(x, a1*x .+ b1,label="l2 with outliers")
plot(x, a3*x .+ b3,label="l1 with outliers")
# plot(x, a5*x .+ b5,label="Huber with outliers")
legend(loc ="best")
ylabel("y")
xlabel("x")
;
```



c)

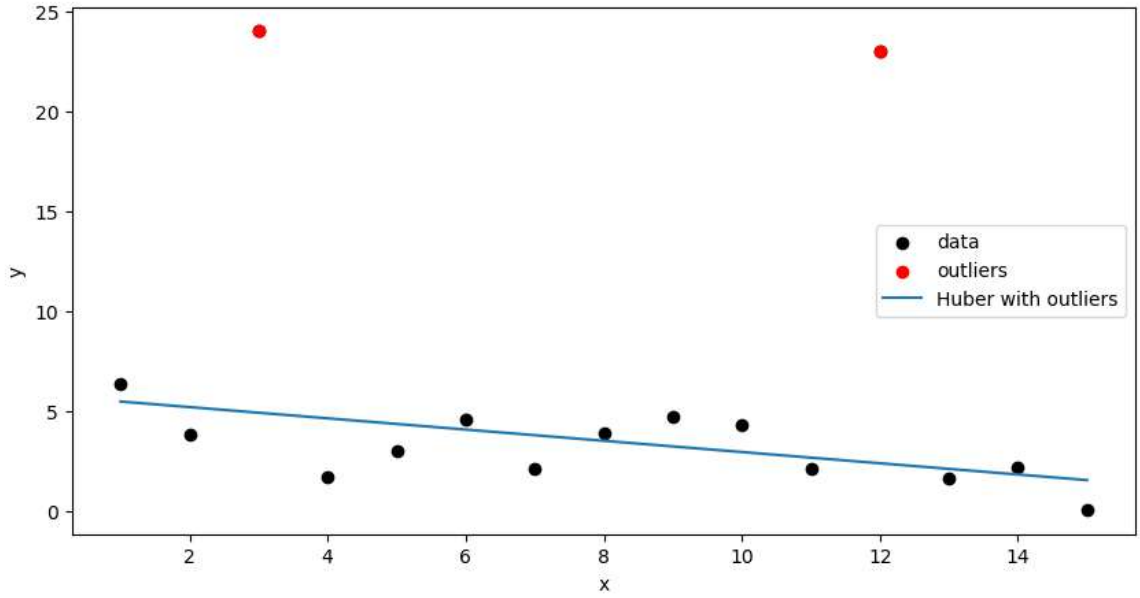
In [28]:
```
m = Model(Ipopt.Optimizer)
M = 1

@variable(m, a5)
@variable(m, b5)
@variable(m, w[x] .<= M)
@variable(m, v[x] .>= 0)
@variable(m, t[x] >= 0)

@expression(m, pred[i in x], a5 * i + b5)
@constraint(m, -t .<= pred - y)
@constraint(m, pred - y .<= t)

@constraint(m, t .<= w + v)

@objective(m, Min, sum(w[i]^2 + 2 * M * v[i] for i in x))

optimize!(m)
```

```
******************************************************************************
This program contains Ipopt, a library for large-scale nonlinear optimization.
 Ipopt is released as open source code under the Eclipse Public License (EPL).
         For more information visit https://github.com/coin-or/Ipopt
******************************************************************************

This is Ipopt version 3.14.4, running with linear solver MUMPS 5.5.1.

Number of nonzeros in equality constraint Jacobian...:        0
Number of nonzeros in inequality constraint Jacobian.:      135
Number of nonzeros in Lagrangian Hessian.............:       15

Total number of variables............................:       47
                     variables with only lower bounds:       30
                variables with lower and upper bounds:        0
                     variables with only upper bounds:       15
Total number of equality constraints.................:        0
Total number of inequality constraints...............:       45
        inequality constraints with only lower bounds:        0
   inequality constraints with lower and upper bounds:        0
        inequality constraints with only upper bounds:       45

iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du alpha_pr  l
s
   0  2.9999970e-01 2.40e+01 6.00e-01  -1.0 0.00e+00    -  0.00e+00 0.00e+00
0
   1  3.2936248e-01 2.39e+01 7.03e-01  -1.0 2.04e+01    -  1.01e-02 1.12e-02h
1
   2  4.2662057e-01 2.37e+01 1.94e+00  -1.0 2.02e+01    -  1.06e-03 9.57e-03f
1
   3  1.1973574e+00 2.32e+01 2.00e+00  -1.0 2.06e+01    -  9.10e-03 2.33e-02h
1
   4  3.2097257e+00 2.24e+01 3.32e+00  -1.0 1.99e+01    -  5.83e-03 3.15e-02h
1
   5  1.1450170e+01 2.00e+01 4.22e+00  -1.0 1.97e+01    -  1.13e-02 1.08e-01h
1
   6  2.4528835e+01 1.63e+01 3.42e+00  -1.0 1.76e+01    -  1.40e-01 1.84e-01h
1
   7  2.9544931e+01 1.51e+01 3.13e+00  -1.0 1.50e+01    -  2.25e-02 7.55e-02h
1
   8  4.3401131e+01 1.16e+01 2.36e+00  -1.0 1.33e+01    -  1.32e-01 2.28e-01h
1
   9  6.1699864e+01 7.60e+00 1.46e+00  -1.0 1.04e+01    -  1.25e-01 3.46e-01h
1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du alpha_pr  l
s
  10  8.2639402e+01 3.24e+00 6.88e-01  -1.0 6.77e+00    -  3.22e-01 5.70e-01h
1
  11  9.9879552e+01 0.00e+00 5.66e-01  -1.0 2.82e+00    -  4.99e-01 1.00e+00h
1
  12  9.9750875e+01 0.00e+00 1.29e-01  -1.0 5.47e-01    -  7.72e-01 1.00e+00f
1
  13  9.9832905e+01 0.00e+00 1.00e-06  -1.0 5.85e-01    -  1.00e+00 1.00e+00f
1
  14  9.6390393e+01 0.00e+00 4.32e-02  -2.5 2.66e-01    -  9.61e-01 7.88e-01f
1
  15  9.5656653e+01 0.00e+00 2.83e-08  -2.5 1.38e-01    -  1.00e+00 1.00e+00f
1
  16  9.5516457e+01 0.00e+00 1.50e-09  -3.8 5.35e-02    -  1.00e+00 1.00e+00f
1
```

```
  17  9.5508268e+01 0.00e+00 1.50e-09  -3.8 1.84e-02    -  1.00e+00 1.00e+00f
1
  18  9.5500579e+01 0.00e+00 1.84e-11  -5.7 9.77e-03    -  1.00e+00 1.00e+00f
1
  19  9.5500001e+01 0.00e+00 1.84e-11  -5.7 4.65e-03    -  1.00e+00 1.00e+00f
1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du alpha_pr  l
s
  20  9.5499856e+01 0.00e+00 1.84e-11  -5.7 2.28e-03    -  1.00e+00 1.00e+00f
1
  21  9.5499749e+01 0.00e+00 2.53e-14  -8.6 1.24e-03    -  1.00e+00 1.00e+00f
1
  22  9.5499739e+01 0.00e+00 2.84e-14  -8.6 6.17e-04    -  1.00e+00 1.00e+00f
1
  23  9.5499736e+01 0.00e+00 2.53e-14  -8.6 3.08e-04    -  1.00e+00 1.00e+00f
1
  24  9.5499736e+01 0.00e+00 2.52e-14  -8.6 1.53e-04    -  1.00e+00 1.00e+00f
1
  25  9.5499736e+01 0.00e+00 2.52e-14  -8.6 7.46e-05    -  1.00e+00 1.00e+00h
1
  26  9.5499736e+01 0.00e+00 1.07e-14  -9.0 3.85e-05    -  1.00e+00 1.00e+00h
1

Number of Iterations....: 26

                                   (scaled)                 (unscaled)
Objective...............:   9.5499735609784139e+01    9.5499735609784139e+01
Dual infeasibility......:   1.0658141036401503e-14    1.0658141036401503e-14
Constraint violation....:   0.0000000000000000e+00    0.0000000000000000e+00
Variable bound violation:   9.2814745774690440e-09    9.2814745774690440e-09
Complementarity.........:   3.8759298670639868e-09    3.8759298670639868e-09
Overall NLP error.......:   3.8759298670639868e-09    3.8759298670639868e-09


Number of objective function evaluations             = 27
Number of objective gradient evaluations             = 27
Number of equality constraint evaluations            = 0
Number of inequality constraint evaluations          = 27
Number of equality constraint Jacobian evaluations   = 0
Number of inequality constraint Jacobian evaluations = 1
Number of Lagrangian Hessian evaluations             = 1
Total seconds in IPOPT                               = 0.226

EXIT: Optimal Solution Found.
```

In [29]:

```
a5 = value(a5)
b5 = value(b5)

using PyPlot
figure(figsize = (10,5))
scatter(x,y,label="data",color = "black")
scatter([x[3],x[12]],[y[3],y[12]],label="outliers",color = "red") # outliers
# plot(x, a1*x .+ b1,label="l2 with outliers")
# plot(x, a3*x .+ b3,label="l1 with outliers")
plot(x, a5*x .+ b5,label="Huber with outliers")
legend(loc ="best")
ylabel("y")
xlabel("x")
;
```



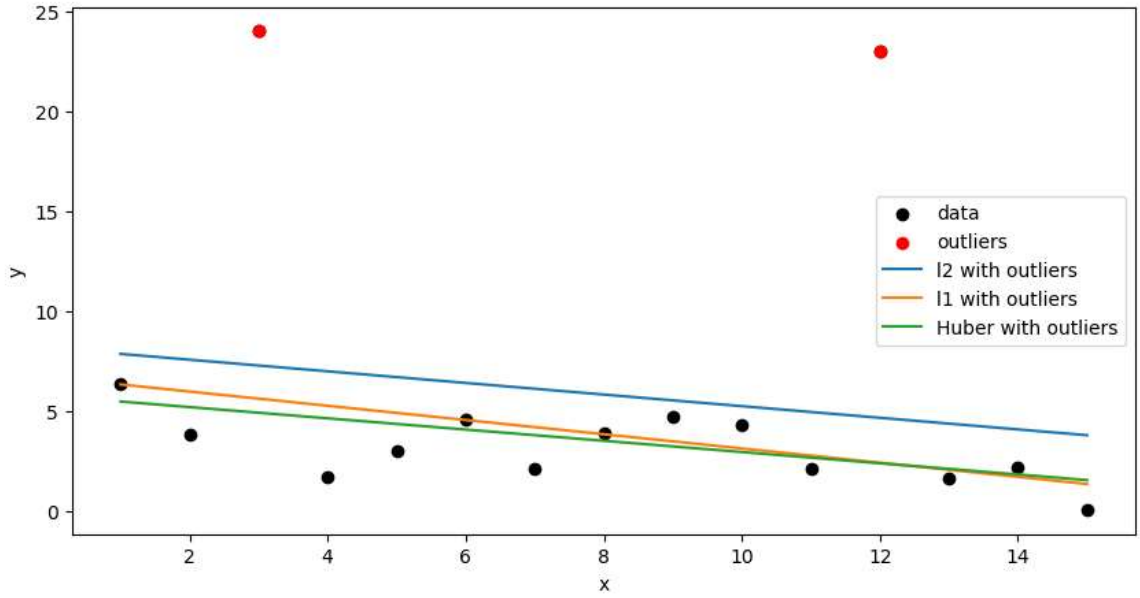
In [30]:
```
println("a = ", a5)
println("b = ", b5)
```

```
a = -0.2811079890443627
b = 5.738120548352265
```

d)

The Huber loss technique gives the best fit to the non-outlier data.

In [75]:
```
# Make all the plots

# fill in random numbers - replace with the values of  (a,b) found in parts (a),
# a1,  a3, a5,  b1, b3, b5 = rand(1,6);

using PyPlot
figure(figsize = (10,5))
scatter(x,y,label="data",color = "black")
scatter([x[3],x[12]],[y[3],y[12]],label="outliers",color = "red") # outliers
plot(x, a1*x .+ b1,label="l2 with outliers")
plot(x, a3*x .+ b3,label="l1 with outliers")
plot(x, a5*x .+ b5,label="Huber with outliers")
legend(loc ="best")
ylabel("y")
xlabel("x")
;
```



3. Heat pipe design

geometric program:

$$\max_{r,T,w} \quad \alpha_4 T r^2$$

$$s.t. \quad \alpha_1 T r/w + \alpha_2 r + \alpha_3 r w \leq C_{max}$$

$$T_{min} \leq T \leq T_{max}$$

$$r_{min} \leq r \leq r_{max}$$

$$w_{min} \leq w \leq w_{max}$$

$$w \leq 0.1 r$$

convex program:

$$\max_{r,T,w} \quad log(\alpha_4) + x + 2y$$

$$s.t. \quad log(e^{log(\alpha_1)+x+y-z} + e^{log(\alpha_2)+y} + e^{log(\alpha_3)+y+z}) \leq log(C_{max})$$

$$log(T_{min}) \leq x \leq log(T_{max})$$

$$log(r_{min}) \leq y \leq log(r_{max})$$

$$log(w_{min}) \leq z \leq log(w_{max})$$

$$z \leq log(0.1) + y$$

$$x := log\, T, y := log\, r, z := log\, w$$

In [6]:
```
m = Model(Ipopt.Optimizer)

@variable(m, x)
@variable(m, y)
@variable(m, z)
```

```
@NLconstraint(m, z <= log(0.1) + y)
@NLconstraint(m, log(exp(x+y-z) + exp(y) + exp(y+z)) <= log(500))

@objective(m, Max, x + 2y)

optimize!(m)

println("T: ", exp(value(x)))
println("r: ", exp(value(y)))
println("w: ", exp(value(z)))
println("objective value: ", exp(objective_value(m)))
```

```
This is Ipopt version 3.14.4, running with linear solver MUMPS 5.5.1.

Number of nonzeros in equality constraint Jacobian...:        0
Number of nonzeros in inequality constraint Jacobian.:        5
Number of nonzeros in Lagrangian Hessian.............:        6

Total number of variables............................:        3
                     variables with only lower bounds:        0
                variables with lower and upper bounds:        0
                     variables with only upper bounds:        0
Total number of equality constraints.................:        0
Total number of inequality constraints...............:        2
        inequality constraints with only lower bounds:        0
   inequality constraints with lower and upper bounds:        0
        inequality constraints with only upper bounds:        2

iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du alpha_pr  l
s
   0  0.0000000e+00 2.30e+00 7.71e-01  -1.0 0.00e+00    -  0.00e+00 0.00e+00
0
   1  1.1490496e+01 8.32e-01 3.67e-01  -1.0 8.32e+00    -  7.31e-01 6.09e-01f
1
   2  1.0981404e+01 1.16e-01 6.99e-02  -1.7 2.06e+00    -  1.00e+00 8.46e-01h
1
   3  1.0840692e+01 4.91e-03 5.26e-03  -2.5 9.08e-01    -  1.00e+00 1.00e+00h
1
   4  1.0845264e+01 1.60e-04 2.21e-04  -3.8 1.46e-01    -  1.00e+00 1.00e+00h
1
   5  1.0845558e+01 0.00e+00 4.31e-07  -5.7 6.57e-03    -  1.00e+00 1.00e+00h
1
   6  1.0845561e+01 0.00e+00 7.08e-12  -8.6 2.59e-05    -  1.00e+00 1.00e+00h
1

Number of Iterations....: 6

                                   (scaled)                 (unscaled)
Objective...............:  -1.0845561178123496e+01    1.0845561178123496e+01
Dual infeasibility......:   7.0848882316454365e-12    7.0848882316454365e-12
Constraint violation....:   0.0000000000000000e+00    0.0000000000000000e+00
Variable bound violation:   0.0000000000000000e+00    0.0000000000000000e+00
Complementarity.........:   2.5250834069909082e-09    2.5250834069909082e-09
Overall NLP error.......:   2.5250834069909082e-09    2.5250834069909082e-09


Number of objective function evaluations             = 7
Number of objective gradient evaluations             = 7
Number of equality constraint evaluations            = 0
Number of inequality constraint evaluations          = 7
Number of equality constraint Jacobian evaluations   = 0
Number of inequality constraint Jacobian evaluations = 7
Number of Lagrangian Hessian evaluations             = 6
Total seconds in IPOPT                               = 0.005

EXIT: Optimal Solution Found.
T: 23.840239130821068
r: 46.39042836246195
w: 4.639042762239439
objective value: 51305.907379387674
```

T: 23.840239130821068

r: 46.39042836246195

w: 4.639042762239439

objective value: 51305.907379387674