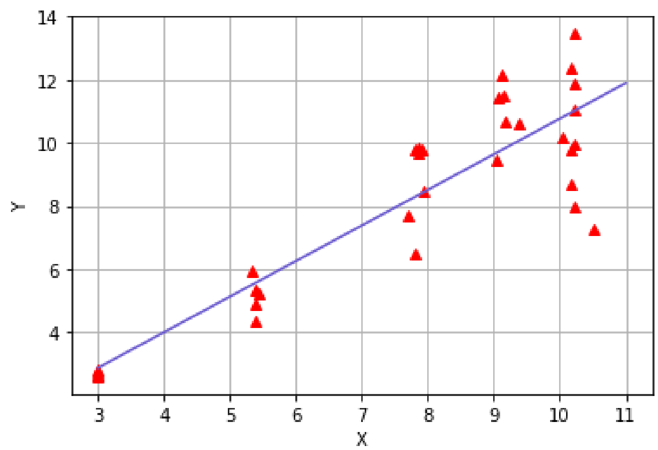
Modelos Estadísticos I **(Fecha: 10/03/2021)**

Tarea #4

Estudiante: Roberto Vásquez Martínez *NUA:* 424662

Problema 1

(a) A continuación mostramos el resultado de la regresión haciendo mínimos cuadrados ordinarios



Y los estimadores de mínimos cuadrados del interceto β_0 y la pendiente β_1 son

$$\hat{\beta}_0 = -0.5059, \quad \text{y} \quad \hat{\beta}_1 = 1.1272,$$

y la varianza de estas estimaciones es

$$V(\hat{\beta}_0) = 0.6907, \quad \text{y} \quad V(\hat{\beta}_1) = 0.0105$$

(b) A continuación mostramos la gráfica de residuos (eje vertical) y valores ajustados (eje horizontal).

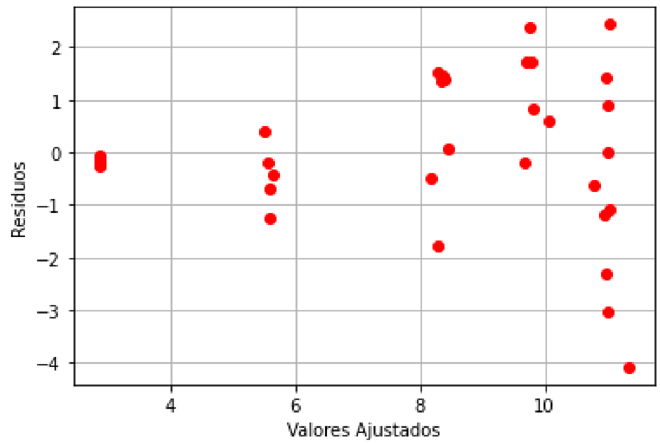


Figura 1: Residuos contra Valores Ajustados

Aquí se observa que la incertidumbre de los errores crece con la respuesta media, por lo que la varianza no puede ser constante.
(c) Hacemos el mismo ejercicio pero ahora graficando residuos (eje vertical) y valores de la variable independiente (eje horizontal), observamos el mismo comportamiento, la dispersión de los errores crece con la variable independiente.

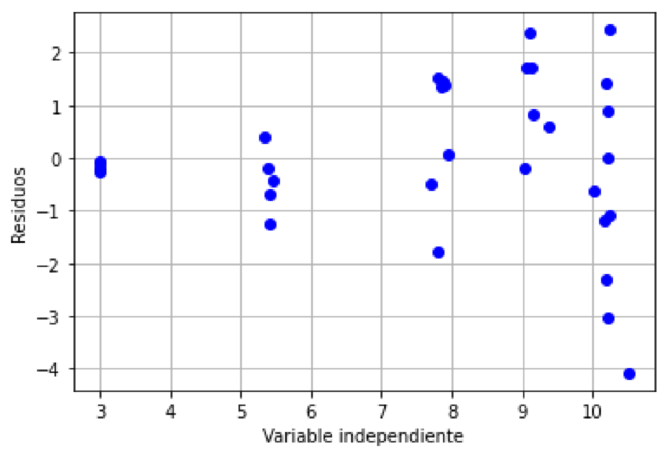


Figura 2: Residuos contra Variable Independiente

(d) Dado que de los incisos anteriores deducimos la varianza no es constante y crece con el valor de la respuesta media así como de la variable independiente nos interesa ver que forma tiene.

Evaluaremos una forma lineal, cuadrática o cúbico, es decir, supondremos que la varianza de la i-ésima σ_i^2 sea de las formas

$$\sigma_i^2 = kX_i^n \quad n = 1, 2, 3. \tag{1.1}$$

La forma en que lo hacemos es evaluar la varianza de las Y's en cada grupo y la media de las X's de cada grupo, para calcular la constante de proporcionalidad hacemos una regresión lineal con esos 5 datos, y comparamos con la gráfica de las funciones en (1.1) para cada $n = 1, 2, 3$.

Mostramos el código y la gráfica correspondiente.

```
# inciso d)

# Medias por grupo
Xmean=np.array([np.mean(X[0:5]),np.mean(X[5:10]),np.mean(X[10:17]),np.mean(X[17
    :23]),np.mean(X[23:])])
# Varianzas por grupo
Yvar=np.array([np.var(Y[0:5],ddof=1),np.var(Y[5:10],ddof=1),np.var(Y[10:17],
    ddof=1),np.var(Y[17:23],ddof=1),np.var(Y[23:],ddof=1)])

# Pendiente regresion por el origen
b1_RO=lambda X,Y: sum(X*Y)/sum(X*X)

figD=plt.figure()
ax=figD.add_subplot(111)
domX=np.arange(min(X),max(X)+1)
ax.plot(domX,domX*b1_RO(Xmean,Yvar),label='Lineal')
ax.plot(domX,(domX**2)*b1_RO(Xmean**2,Yvar),label='Cuadratico')
ax.plot(domX,(domX**3)*b1_RO(Xmean**3,Yvar),label='Cubico')
plt.plot(Xmean,Yvar,'ob')
plt.legend()
plt.grid(True)
plt.show()
```

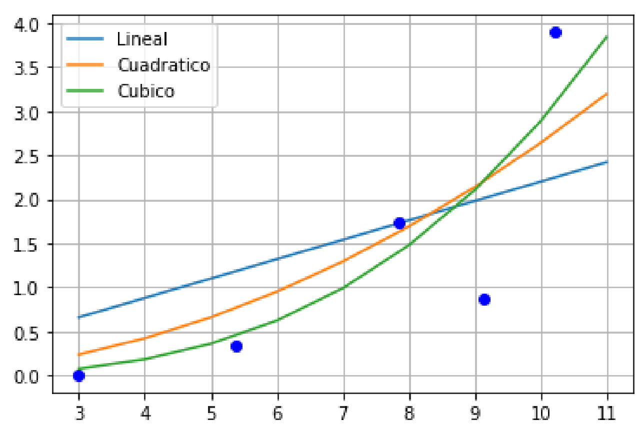


Figura 3: Forma de las varianzas

Podemos ver graficamente que la mejor forma según esta gráfica sería la cúbica, luego asignamos la varianza de cada observación como $\sigma_i^2 = kX_i^3$.

(e) Según el inciso anterior asignamos los pesos como $w_i = \sigma_i^{-2}$ para cada observación.

El código de esta parte es

```
# Asignar pesos cubicos
# inciso e)
sigma2=b1_RO(Xmean**3,Yvar)*(X**3)
W=1.0/sigma2
invV=np.diag(W)
```

Estos pesos los usamos para hacer la regresión lineal por Mínimos Cuadrados Ponderados en el siguiente inciso.

(f) Con los datos hasta el momento la regresión lineal por mínimos cuadrados ponderados nos da el siguiente resultado.

```
# Regresi n (M nimos cuadrados ponderados)
# inciso f)
Xreg=np.array([np.ones(len(Y)),X]).T
Yreg=np.array([Y]).T
beta_MCP=np.linalg.inv(Xreg.T@invV@Xreg)@(Xreg.T@invV@Yreg)
Var_beta_MCP=np.linalg.inv(Xreg.T@invV@Xreg)*((Yreg-Xreg@beta_MCP).T@invV@(Yreg
    -Xreg@beta_MCP))/(len(X)-2)
print("\n\nMODELO REGRESION SIMPLE (Minimos Cuadrados Ponderados)")

print("\nLos coeficientes son b0=%.4f y b1=%.4f" %(beta_MCP[0],beta_MCP[1]))

print("\nLas varianzas V(b0)=%.4f, V(b1)=%.4f" %(Var_beta_MCP[0][0],
    Var_beta_MCP[1][1]))

figF=plt.figure()
ax=figF.add_subplot(111)
ax.plot(X,Y,'^r')
domX=np.arange(min(X),max(X)+1)
ax.plot(domX,estY(domX,beta_MCP[0],beta_MCP[1]),color='slateblue')
plt.xlabel("X")
plt.ylabel("Y")
plt.grid(True)
plt.show()
```

En este caso tenemos que las estimaciones del intercepto y la pendiente son

$$\hat{\beta}_0 = -0.8363 \quad \text{y} \quad \hat{\beta}_1 = 1.1728.$$

Y las varianzas estimadas correspondientes son

$$V(\hat{\beta}_0) = 0.0609 \quad \text{y} \quad V(\hat{\beta}_1) = 0.0030.$$

Si bien las estimaciones de β_0 y β_1 no difieren tanto de las hechas en a) si lo hacen las varianzas, lo que muestra que las estimaciones en el caso de Mínimos Cuadrados Ponderados son de menor varianza que las de Mínimos Cuadrados Ordinarios, en este caso donde las varianzas no son constantes.

□

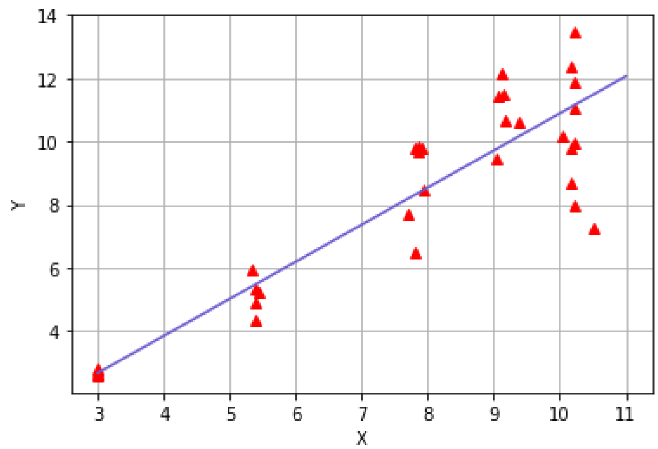


Figura 4: Regresión Lineal por Mínimos Cuadrados Ponderados

Problema 2

En esta parte mostramos el código así como la salida de cada una de las cuentas matriciales para obtener los resultados de cada parte de *i*) al *v*) del Teorema visto en la clase del 3 de Marzo de 2021.

(a)

```
##############
# PROBLEMA 2 #
##############
datosP2=pd.read_csv('DatosP2.csv')
X0=datosP2['X0'].to_numpy()
X1=datosP2['X1'].to_numpy()
X2=datosP2['X2'].to_numpy()
Z1=datosP2['Z1'].to_numpy()
Z2=datosP2['Z2'].to_numpy()
Y_P2=datosP2['Y'].to_numpy()
np.set_printoptions(formatter={'float': lambda x: "{0:0.4f}".format(x)}) #
    Imprimir 4 decimales
# inciso a)

# Matrices de diseno
Xdis=np.array([X0,X1,X2]).T
Zdis=np.array([Z1,Z2]).T
Ydis=np.array([Y_P2]).T

# Matriz completa
Wdis=np.array([X0,X1,X2,Z1,Z2]).T

# Proyeccion ortogonal
R=np.identity(np.shape(Xdis)[0])-Xdis@(np.linalg.inv(Xdis.T@Xdis))@Xdis.T
RG=np.identity(np.shape(Wdis)[0])-Wdis@(np.linalg.inv(Wdis.T@Wdis))@Wdis.T
L=np.linalg.inv(Xdis.T@Xdis)@Xdis.T@Zdis
M=np.linalg.inv(Zdis.T@R@Zdis)
```

(b) Con las cuentas hechas en a) mostramos el código y el resultado para cada inciso.

i)
```
    # parte i)
    beta_G=hat_beta-L@gamma_G
    print("La expresion para beta_G es:\n")
    print(beta_G)
```

```
La expresion para beta_G es:
[[-8.4952]
[1.6721]
[1.1769]]
```

ii)
```
     # parte ii)
     gamma_G=M@Zdis.T@R@Ydis
hat_beta=np.linalg.inv(Xdis.T@Xdis)@Xdis.T@Ydis
print("La expresion para gamma_G es:\n")
print(gamma_G)
```

```
La expresion para gamma_G es:
[[0.5307]
[0.3649]]
```

iii)
```
     # parte iii)
YTR_GY_Exp1=(Ydis-Zdis@gamma_G).T@R@(Ydis-Zdis@gamma_G)
print("La primera expresion para YTR_GY es:\n")
print(YTR_GY_Exp1)
```

```
La primera expresion para YTR_GY es:
[[84.8917]]
```

iv)
```
     # parte iv)
YTR_GY_Exp2=Ydis.T@R@Ydis-gamma_G.T@Zdis.T@R@Ydis
print("La segunda expresion para YTR_GY es:\n")
print(YTR_GY_Exp2)
```

```
La segunda expresion para YTR_GY es:
[[84.8917]]
```

v)

```
    # parte v)
delta_G=np.linalg.inv(Wdis.T@Wdis)@Wdis.T@Ydis
print("La estimacion de MC de delta_G es:\n")
print(delta_G)
s2_G=(((Ydis-Wdis@delta_G).T@(Ydis-Wdis@delta_G))/(len(Y_P2)-5))[0][0]
print("\nEstimacion de la varianza:\n ")
print("%.4f" %(s2_G))
# Varianza del estimador delta_G
block11=np.linalg.inv(Xdis.T@Xdis)+L@M.T@L.T
block12=(-1)*(L@M)
block21=block12.T
block22=M

Var_delta_G=np.block([[block11,block12],[block21,block22]])
print("La matriz de covarianzas del estimador delta_G es:\n")
print(Var_delta_G)
```

```
La matriz de covarianzas del estimador delta_G es:
[[1.1835 -0.0370 -0.0334 -0.0487 -0.0142]
[-0.0370 0.0065 0.0004 0.0007 -0.0004]
[-0.0334 0.0004 0.0033 0.0013 -0.0005]
[-0.0487 0.0007 0.0013 0.0030 0.0002]
[-0.0142 -0.0004 -0.0005 0.0002 0.0013]]
```

□