Caracterización de una Nueva Función de Densidad Bivariada: Propiedades y Estimación de Parámetros

Función de Densidad Bivariada

In[•]:=
```
(*Parámetros de prueba*){a1, b1, c1, d1} = {2.2, 2.2, 2.2, 2.2};
```

Constante de Normalización

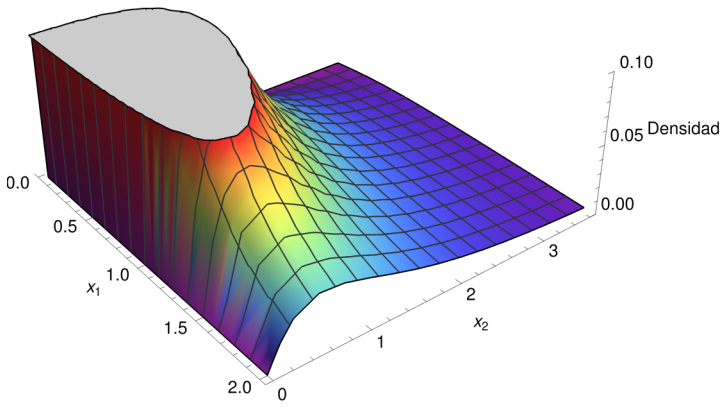
In[3]:=
```
K[a_, b_, c_, d_] := Beta[a, b] * Beta[c, d];
                     beta       beta
```

Superficie de la función de densidad

In[4]:=
$$f[x\_, y\_] := \frac{1}{K[a, b, c, d]} \frac{x^{a-1}\, y^{b-1}\, (x+y)^{d-(a+b)}}{(x+y+1)^{c+d}};$$

In[18]:=
```
Block[{a = a1, b = b1, c = c1, d = d1},
 Plot3D[f[x, y], {x, 0, 2}, {y, 0, 3.6}, PlotRange → {0, 10^-1},
  AxesLabel → {x1, x2, Densidad}, ColorFunction → "Rainbow", Boxed → False]]
```
bloquea · representación gráfica 3D · rango de representación · etiqueta de ejes · función de color · rodeado··· · falso

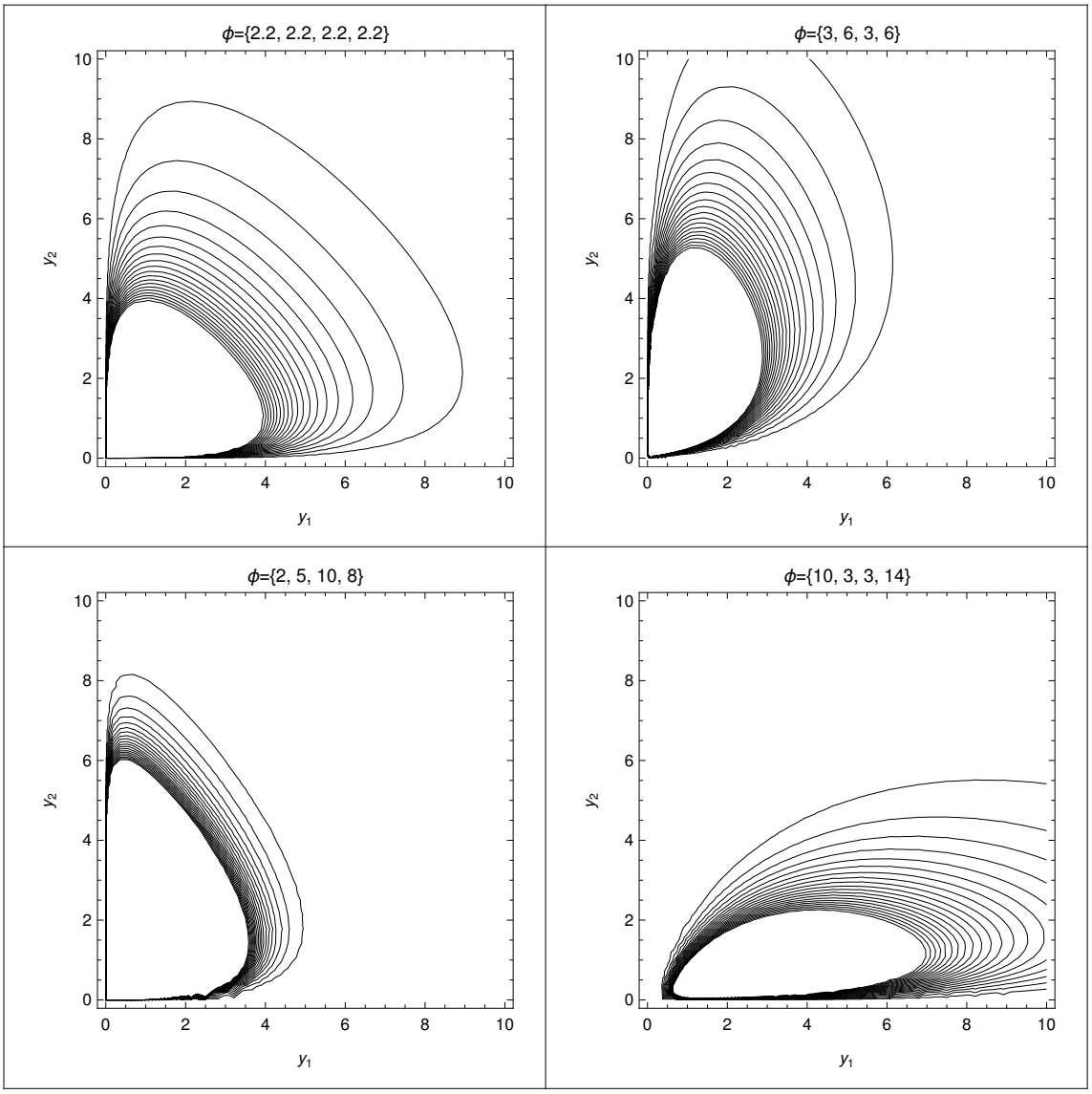
Out[18]=



Curvas de Nivel Para La Función De Densidad Conjunta

```
f[x_, y_, a_, b_, c_, d_] :=
 1 / K[a, b, c, d] (x^(a - 1) y^(b - 1) (x + y)^(d - (a + b))) / (x + y + 1)^(c + d)
(*Define los cuatro conjuntos de parámetros*)
params = {{2.2, 2.2, 2.2, 2.2}, {3, 6, 3, 6}, {2, 5, 10, 8}, {10, 3, 3, 14}};
(*Genera las curvas de nivel para cada conjunto de parámetros*)
contourPlots =
  ContourPlot[#, {x, 0, 10}, {y, 0, 10}, ContourShading → None, Contours → 20,
   FrameLabel → {y1, y2}, PlotLabel → Style["ϕ=" <> ToString[#2], 10]] & @@@
  ({f[x, y, ###], {###}} & @@@ params);
(*Organiza las gráficas en una cuadrícula*)
GraphicsGrid[Partition[contourPlots, 2], Spacings → {20, 20}, Frame → All]
```
representación de contornos · sombreado de contornos · ning··· · contornos · etiqueta de marco · etiqueta de rep·· · estilo · convierte a cadena de caracteres · rejilla de gráficos · particiona · espaciados · marco · todo

Out[•]=



Distribución de Probabilidad Acumulada Marginal para Y_1 (azul) y Y_2(roja)

In[•]:=
```
ClearAll[Fx, Fy, generatePlotx, plots, generatePloty, combinedPlot];
```
borra todo

```
LaunchKernels[7];
```
lanza kernels

```
Fx[x_, a_, b_, c_, d_] := N[Block[{a2 = a, b2 = b, c2 = c, d2 = d},
```
bloquea

$$\frac{x^{d2}}{d2\ \text{Beta}[a2, b2]\ \text{Beta}[c2, d2]}\ \text{NIntegrate}\left[v^{-1-d2+a2}\ (1+v)^{d2-a2-b2}\right]$$
integra numéricamente

```
        Hypergeometric2F1[d2, c2 + d2, 1 + d2, -((1 + 1/v) x)], {v, 0, Infinity}]], 3];
```
2F1 hipergeométrica · infinito

```
Fy[y_, a_, b_, c_, d_] := N[Block[{a2 = a, b2 = b, c2 = c, d2 = d},
```
· bloquea

```
  y^d2/(d2 Beta[a2, b2] Beta[c2, d2]) NIntegrate[v^(-1+a2) (1 + v)^(d2-a2-b2)
```
integra numéricamente

```
        Hypergeometric2F1[d2, c2 + d2, 1 + d2, -((1 + v) y)], {v, 0, Infinity}]], 3];
```
2F1 hipergeométrica · infinito

```
DistributeDefinitions[Fx, Fy];
```
distribuye definiciones

```
generatePlotx[a_, b_, c_, d_] := Module[{pointsx, label},
```
módulo

```
  pointsx = ParallelTable[{x, Fx[x, a, b, c, d]}, {x, 0.1, 12, 0.1}];
```
tabla en paralelo

```
  label = "ϕ = (" <> ToString[a] <> ", " <>
```
convierte a cadena de caracteres

```
    ToString[b] <> ", " <> ToString[c] <> ", " <> ToString[d] <> ")";
```
convierte a cadena de cara·· · convierte a cadena de cara·· · convierte a cadena de caracteres

```
  ListPlot[pointsx, Joined → True,
```
representación de lista · unido · verdadero

```
    PlotRange → {{0, 12}, {0, 1}}, PlotLabel → label, PlotStyle → Blue];
```
rango de representación · etiqueta de representación · estilo de repre··· · azul

```
generatePloty[a_, b_, c_, d_] := Module[{pointsy, label},
```
módulo

```
  pointsy = ParallelTable[{y, Fy[y, a, b, c, d]}, {y, 0.1, 12, 0.1}];
```
tabla en paralelo

```
  label = "ϕ = (" <> ToString[a] <> ", " <>
```
convierte a cadena de caracteres

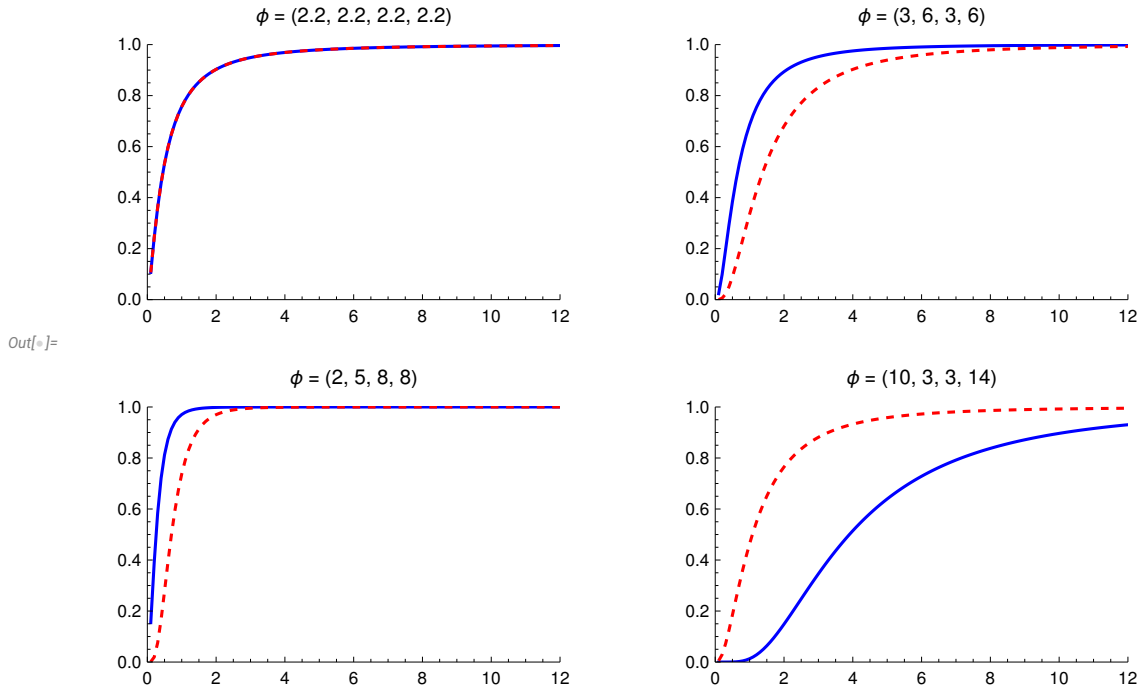
```
    ToString[b] <> ", " <> ToString[c] <> ", " <> ToString[d] <> ")";
```
convierte a cadena de cara·· · convierte a cadena de cara·· · convierte a cadena de caracteres

```
  ListPlot[pointsy, Joined → True, PlotRange → {{0, 12}, {0, 1}},
```
representación de lista · unido · verd··· · rango de representación

```
    PlotLabel → label, PlotStyle → Directive[Red, Dashed]];
```
etiqueta de representación · estilo de repre··· · directiva · rojo · rayado

In[•]:=
```
plots = {Show[generatePlotx[2.2, 2.2, 2.2, 2.2], generatePloty[2.2, 2.2, 2.2, 2.2]],
    Show[generatePlotx[3, 6, 3, 6], generatePloty[3, 6, 3, 6]],
    Show[generatePlotx[2, 5, 8, 8], generatePloty[2, 5, 8, 8]],
    Show[generatePlotx[10, 3, 3, 14], generatePloty[10, 3, 3, 14]]};
combinedPlot = GraphicsGrid[Partition[plots, 2], Spacings → {20, 20},
 ImageSize → Large]
(* Añadir la leyenda *)
Legended[combinedPlot, Placed[{"F(y₁)", "F(y₂)"}, Below]];
CloseKernels[];
```

Out[•]=



Función de Supervivencia

In[•]:=
```
(*Parámetros de prueba*){a1, b1, c1, d1} = {2.2, 2.2, 2.2, 2.2};
```

In[6]:=
```
S[x_, y_] := N[Block[{a = a1, b = b1, c = c1, d = d1},  1 / (d Beta[a, b] Beta[c, d])
```
bloquea

$$\left(y^d \int_{\frac{x}{y}}^{\infty} v^{-1+a} (1+v)^{d-a-b}\, \text{Hypergeometric2F1}\left[d, c+d, 1+d, -((1+v)\,y)\right] dv + x^d \int_0^{\infty} v^{-1-d+a} (1+v)^{d-a-b}\, \text{Hypergeometric2F1}\left[d, c+d, 1+d, -\left(\left(1+\frac{1}{v}\right)x\right)\right] dv\right), 3\Big];$$

2F1 hipergeométrica

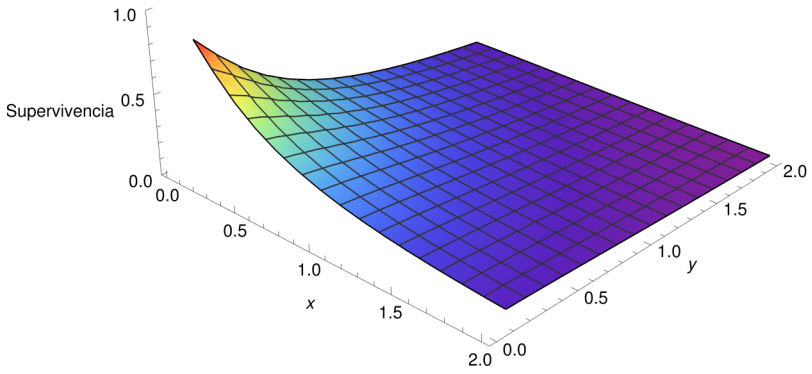
In[13]:=
```
S[x_, y_] := N[Block[{a = a1, b = b1, c = c1, d = d1},
```
bloquea

$$1 - \frac{1}{d\,\text{Beta}[a, b]\,\text{Beta}[c, d]}\left(y^d\, \text{NIntegrate}\left[v^{-1+a} (1+v)^{d-a-b}\, \text{Hypergeometric2F1}\left[d, c+d, 1+d, -((1+v)\,y)\right], \{v, x/y, \text{Infinity}\}\right] + x^d\, \text{NIntegrate}\left[v^{-1-d+a} (1+v)^{d-a-b}\, \text{Hypergeometric2F1}\left[d, c+d, 1+d, -\left(\left(1+\frac{1}{v}\right)x\right)\right], \{v, 0, x/y\}\right]\right), 3\Big];$$

integra numéricamente · 2F1 hipergeométrica · infinito · integra numéricamente · 2F1 hipergeométrica

In[17]:=
```
Plot3D[S[x, y], {x, 0.1, 2}, {y, 0.1, 2}, PlotRange → {0, 1},
 AxesLabel → {x, y, Supervivencia}, ColorFunction → "Rainbow", Boxed → False]
```
representación gráfica 3D · rango de representación · etiqueta de ejes · función de color · rodeado··· · falso

Out[17]=

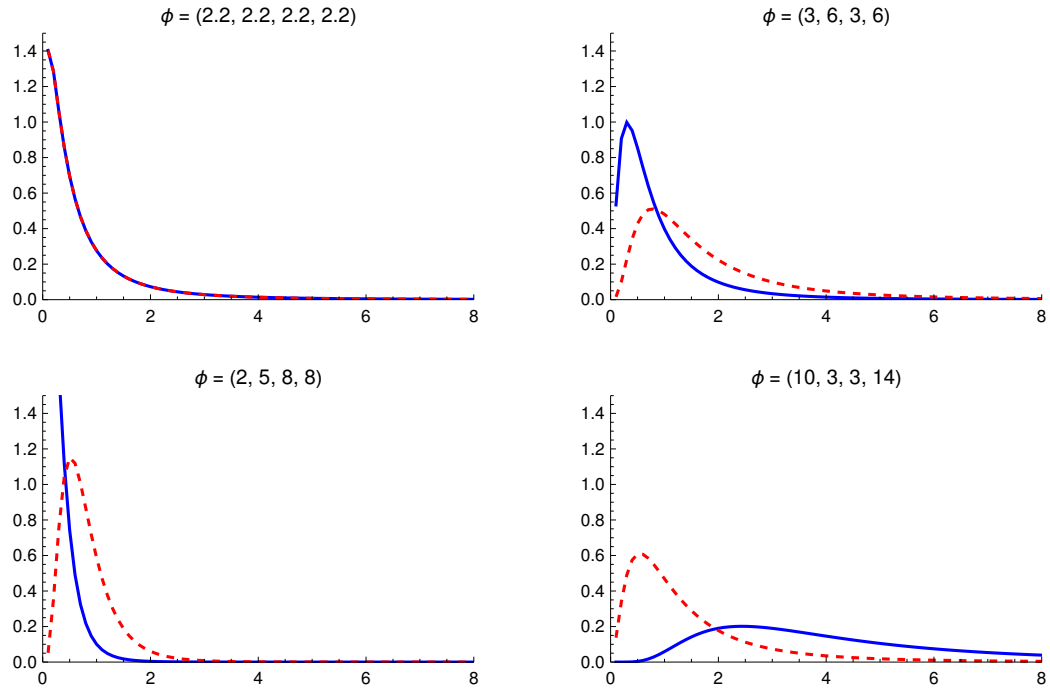


Funciones de Densidad Marginal para Y_1 (azul) y Y_2(roja)

In[•]:=
```
ClearAll[fx, fy, generatePlotx, plots, generatePloty, combinedPlot];
```
borra todo
```
LaunchKernels[7];
```
lanza kernels
```
fx[x_, a_, b_, c_, d_] := N[Block[{a2 = a, b2 = b, c2 = c, d2 = d},
```
· bloquea
```
  NIntegrate[1/(Beta[a2, b2] Beta[c2, d2]) x^(a2-1) y^(b2-1) (x + y)^(d2-(a2+b2))/(x + y + 1)^(c2+d2), {y, 0, Infinity}]], 3];
```
integra numéricamente · infinito
```
fy[y_, a_, b_, c_, d_] := N[Block[{a2 = a, b2 = b, c2 = c, d2 = d},
```
· bloquea
```
  NIntegrate[1/(Beta[a2, b2] Beta[c2, d2]) x^(a2-1) y^(b2-1) (x + y)^(d2-(a2+b2))/(x + y + 1)^(c2+d2), {x, 0, Infinity}]], 3];
```
integra numéricamente · infinito
```
DistributeDefinitions[fx, fy];
```
distribuye definiciones
```
generatePlotx[a_, b_, c_, d_] := Module[{pointsx, label},
```
módulo
```
  pointsx = ParallelTable[{x, fx[x, a, b, c, d]}, {x, 0.1, 8, 0.1}];
```
tabla en paralelo
```
  label = "ϕ = (" <> ToString[a] <> ", " <>
```
convierte a cadena de caracteres
```
    ToString[b] <> ", " <> ToString[c] <> ", " <> ToString[d] <> ")";
```
convierte a cadena de cara·· · convierte a cadena de cara·· · convierte a cadena de caracteres
```
  ListPlot[pointsx, Joined → True,
```
representación de lista · unido · verdadero
```
   PlotRange → {{0, 8}, {0, 1.5}}, PlotLabel → label, PlotStyle → Blue];
```
rango de representación · etiqueta de representación · estilo de repre··· · azul
```
generatePloty[a_, b_, c_, d_] := Module[{pointsy, label},
```
módulo
```
  pointsy = ParallelTable[{y, fy[y, a, b, c, d]}, {y, 0.1, 8, 0.1}];
```
tabla en paralelo
```
  label = "ϕ = (" <> ToString[a] <> ", " <>
```
convierte a cadena de caracteres
```
    ToString[b] <> ", " <> ToString[c] <> ", " <> ToString[d] <> ")";
```
convierte a cadena de cara·· · convierte a cadena de cara·· · convierte a cadena de caracteres
```
  ListPlot[pointsy, Joined → True, PlotRange → {{0, 8}, {0, 1.5}},
```
representación de lista · unido · verd··· · rango de representación
```
   PlotLabel → label, PlotStyle → Directive[Red, Dashed]];
```
etiqueta de representación · estilo de repre··· · directiva · rojo · rayado

In[•]:=
```
plots = {Show[generatePlotx[2.2, 2.2, 2.2, 2.2], generatePloty[2.2, 2.2, 2.2, 2.2]],
    Show[generatePlotx[3, 6, 3, 6], generatePloty[3, 6, 3, 6]],
    Show[generatePlotx[2, 5, 8, 8], generatePloty[2, 5, 8, 8]],
    Show[generatePlotx[10, 3, 3, 14], generatePloty[10, 3, 3, 14]]};
combinedPlot = GraphicsGrid[Partition[plots, 2], Spacings → {20, 20},
 ImageSize → Large]
CloseKernels[];
```

Out[•]=



In[•]:=
```
Export["/home/llerzy/Documentos/density.png", combinedPlot, "PNG"]
```

Out[•]=
```
/home/llerzy/Documentos/density.png
```

Momentos

In[201]:=
```
(*Definir la función Mom para calcular los momentos*)
Mom[m_, n_, a_, b_, c_, d_] := Beta[c - m - n, m + n + d] Beta[m + a, n + b] / K[a, b, c, d];
(*Definir cada conjunto de parámetros*)
paramSets = {{"{2.2, 2.2, 2.2, 2.2}", 2.2, 2.2, 2.2, 2.2}, {"{3, 6, 3, 6}", 3, 6, 3, 6},
   {"{2, 5, 8, 8}", 2, 5, 8, 8}, {"{10, 3, 3, 14}", 10, 3, 3, 14}};
(*Calcular y almacenar los resultados para cada conjunto de parámetros*)
results = Table[
   Module[{a = param[[2]], b = param[[3]], c = param[[4]], d = param[[5]], meanX, meanY, varianceX,
     varianceY, covarianceXY, moment2X, moment2Y}, meanX = N[Mom[1, 0, a, b, c, d]];
    (*Media de X*)meanY = N[Mom[0, 1, a, b, c, d]];
    (*Media de Y*)moment2X = N[Mom[2, 0, a, b, c, d]];
    (*Momento de segundo orden para X*)moment2Y = N[Mom[0, 2, a, b, c, d]];
    (*Momento de segundo orden para Y*)
    covarianceXY = N[Mom[1, 1, a, b, c, d] - Mom[1, 0, a, b, c, d] × Mom[0, 1, a, b, c, d]];
    (*Covarianza*)varianceX = N[moment2X - meanX^2];
    (*Varianza de X*)varianceY = N[moment2Y - meanY^2];
    (*Varianza de Y*){param[[1]], meanX, meanY, varianceX, varianceY, covarianceXY}],
   {param, paramSets}];
(*Mostrar la tabla*)
TableForm[results,
  TableHeadings → {None, {"Configuración", "E[X]", "E[Y]", "V[X]", "V[Y]", "Cov(X,Y)"}}]
```

Out[204]//TableForm=

Configuración	E[X]	E[Y]	V[X]	V[Y]	Cov(X,Y)
{2.2, 2.2, 2.2, 2.2}	0.916667	0.916667	7.85108	7.85108	5.13503
{3, 6, 3, 6}	1.	2.	1.8	5.8	2.2
{2, 5, 8, 8}	0.326531	0.816327	0.0770512	0.251978	0.0395668
{10, 3, 3, 14}	5.38462	1.61538	34.4675	4.31361	8.60947