In[1]:=
Stability analysis of RM1 and RM2;

Constants:

In[2]:=
```
alpha = p = 2;
beta = q = 1;
c = 1;
g = 1;
d = 2;
f = 2;
```

normalized values to 6022 (meeting the condition z0^(1/q) > x0*a0^f)

In[8]:=
```
x0 = 1;
y0 = 1.5;
z0 = 3.5;
a0 = 1.125;
b0 = 1.5;
```

Cases from Table 4

RM2s

Case 1 : x' = y' = 0, var a, b

Case 2 : y' = a' = 0, var x, b

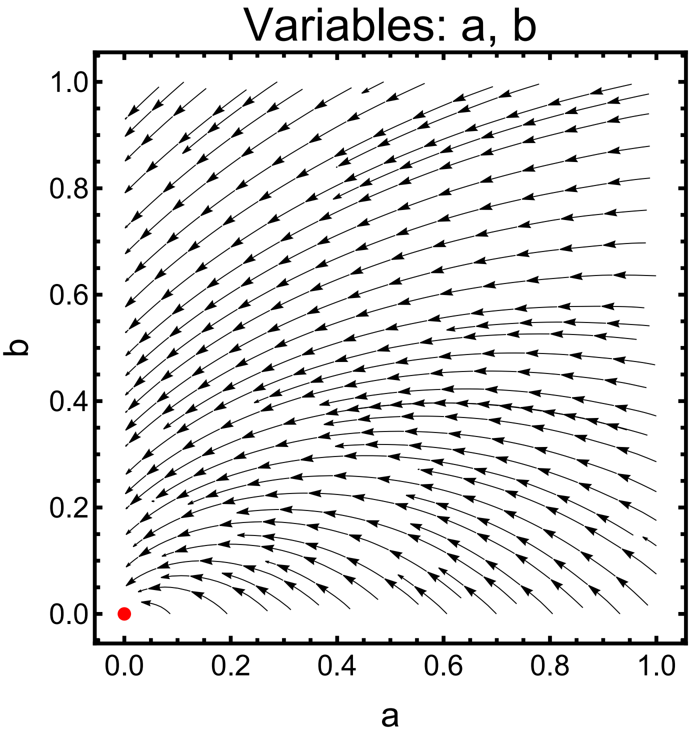
Case 3 : a' = b' = 0, var x, y

Steady States Case 1

In[13]:=
```
RM2sc1 = Solve[{-x*a^f - y*b*a^g == 0, x*a^f - y*b*a^g == 0}, {a, b}]
```

In[14]:=
```
Show[StreamPlot[{-x0 * a^f - y0 * b * a^g, x0 * a^f - y0 * b * a^g},
   {a, 0, 1}, {b, 0, 1}, Frame → True, FrameStyle → Directive[Black, 15],
   FrameLabel → {Style["a", 18], Style["b", 18]}, FrameTicks → Automatic,
   FrameTicksStyle → Directive[Thick, Black, 15], LabelStyle → Directive[Black],
   PlotLabel → Style["Variables: a, b", Large], StreamColorFunction → None,
   StreamStyle → Black], Graphics[{PointSize[Large], Red, Point[{0, 0}]}],
  FrameStyle → {Thick, Directive[Thick, Black]}]
```

Out[14]=



Type of critical point

Calculate the Jacobian

In[15]:=
```
Jc1 = D[{-x * a^f - y * b * a^g, x * a^f - y * b * a^g}, {{a, b}}]
```

Calculate the Jacobian for the found steady states

In[16]:=
```
Jsc1 = Jc1 /. RM2sc1[[2]]
```

Steady States Case 2

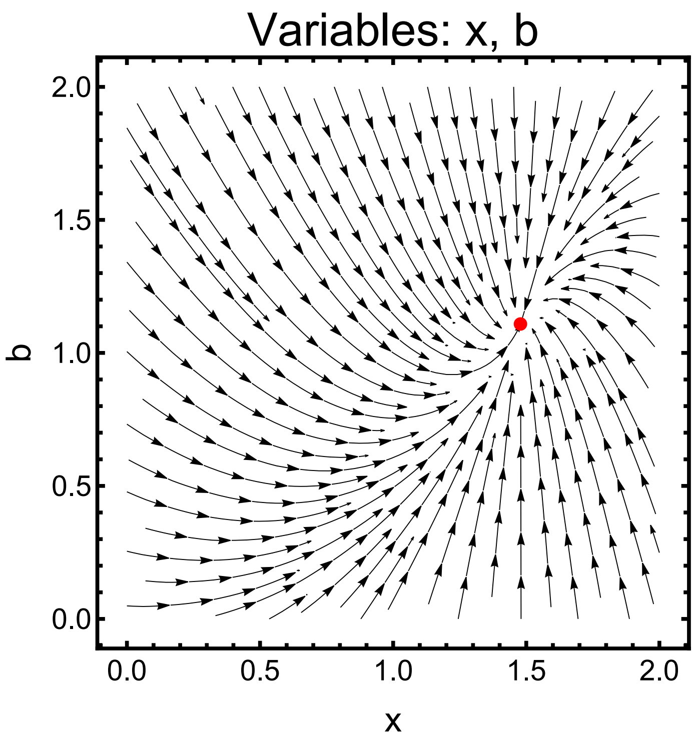
In[17]:=
```
RM2sc2 = Solve[{z^(1/p) - x * a^f == 0, x * a^f - y * b * a^g == 0}, {x, b}]
```

Steady State at given values

In[18]:=
```
solRM2sc2 = RM2sc2[[1]] /. {y → y0, z → z0, a → a0};
xsolc2 = (Part[solRM2sc2, 1] /. Rule → List)[[2]];
bsolc2 = (Part[solRM2sc2, 2] /. Rule → List)[[2]];
```

In[21]:=
```
Show[StreamPlot[{z0^(1/p) - x * a0^f, x * a0^f - y0 * b * a0^g},
   {x, 0, 2}, {b, 0, 2}, Frame → True, FrameStyle → Directive[Black, 18],
   FrameLabel → {Style["x", 18], Style["b", 18]}, FrameTicks → Automatic,
   FrameTicksStyle → Directive[Thick, Black, 15], LabelStyle → Directive[Black],
   PlotLabel → Style["Variables: x, b", Large], StreamColorFunction → None,
   StreamStyle → Black], Graphics[{PointSize[Large], Red, Point[{xsolc2, bsolc2}]}],
  FrameStyle → {Thick, Directive[Thick, Black]}]
```

Out[21]=



Type of critical point

Calculate the Jacobian

In[22]:=
```
Jc2 = D[{z^(1/p) - x * a^f, a^f - y * b * a^g}, {{x, b}}]
```

Calculate the Jacobian for the found steady states

In[23]:=
```
Jc21 = Jc2 /. RM2sc2 [[1]]
```

Eigen value equation (expansion of the characteristic equation)

In[24]:=
```
eve1 = Solve[l^2 - l * Tr[Jc21] + Det[Jc21] == 0, l]
```

Analyse the sign of Tr[J], Det[J] and Tr[J]^2 - 4 Det[J]

In[25]:=
```
Trc2 = Tr[Jc21]
Dc2 = Det[Jc21]
Sc2 = Tr[Jc21]^2 - 4 Det[Jc21]
Simplify[Sc2]
```

Steady States Case 3

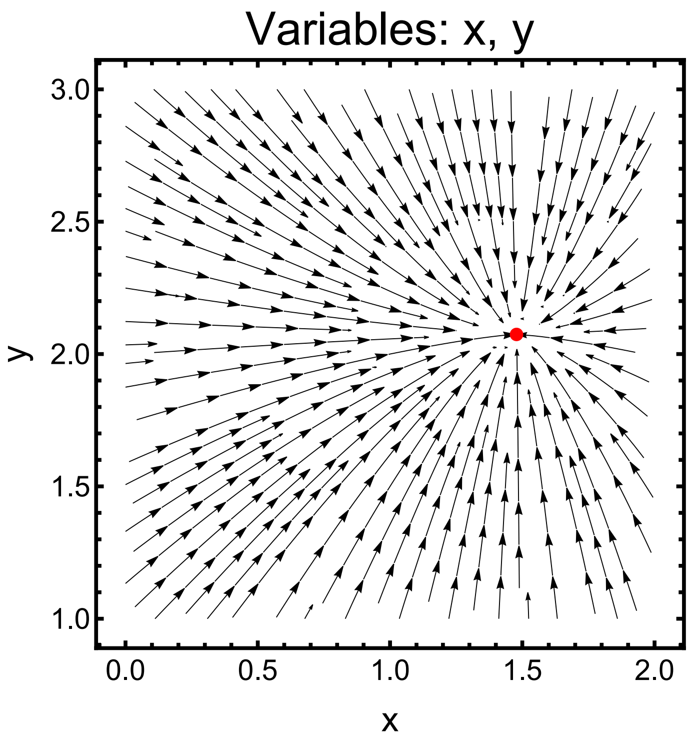
In[29]:=
```
RM2sc3 = Solve[{z^(1/p) - x * a^f == 0, z^(1/q) - y * b * a^g == 0}, {x, y}]
```

Steady State at given values

In[30]:=
```
solRM2sc3 = RM2sc3[[1]] /. {z → z0, a → a0, b → b0};
xsolc3 = (Part[solRM2sc3, 1] /. Rule → List)[[2]];
ysolc3 = (Part[solRM2sc3, 2] /. Rule → List)[[2]];
```

In[33]:=
```
Show[StreamPlot[{z0^(1/p) - x * a0^f, z0^(1/q) - y * b0 * a0^g},
  {x, 0, 2}, {y, 1, 3}, Frame → True, FrameStyle → Directive[Black, 18],
  FrameLabel → {Style["x", 18], Style["y", 18]}, FrameTicks → Automatic,
  FrameTicksStyle → Directive[Thick, Black, 15], LabelStyle → Directive[Black],
  PlotLabel → Style["Variables: x, y", Large], StreamColorFunction → None,
  StreamStyle → Black], Graphics[{PointSize[Large], Red, Point[{xsolc3, ysolc3}]}],
 FrameStyle → {Thick, Directive[Thick, Black]}]
```

Out[33]=



Type of critical point

Calculate the Jacobian

In[34]:=
```
Jc3 = D[{z^(1/p) - x * a^f, z^(1/q) - y * b * a^g}, {{x, y}}]
```

Calculate the Jacobian for the found steady states

In[35]:=
```
Jc31 = Jc3 /. RM2sc3 [[1]]
```

Eigen value equation (expansion of the characteristic equation)

In[36]:=
```
eve3 = Solve[l^2 - l * Tr[Jc31] + Det[Jc31] == 0, l]
```

Analyse the sign of Tr[J], Det[J] and Tr[J]^2 - 4 Det[J]

In[37]:=
```
Trc3 = Tr[Jc31]
Dc3 = Det[Jc31]
Sc3 = Tr[Jc31]^2 - 4 Det[Jc31]
Simplify[Sc3]
```

RM2

Case 4 : x' = y' = 0, var a, b => same as case 1 RM2s
Case 5 : x' = a' = 0, var y, b
Case 6 : y' = a' = 0, var x, b => same as case 2 RM2s
Case 7: a' = b' = 0, var x, y
Case 8: a' = 0, var x, y, b

Steady States Case 5

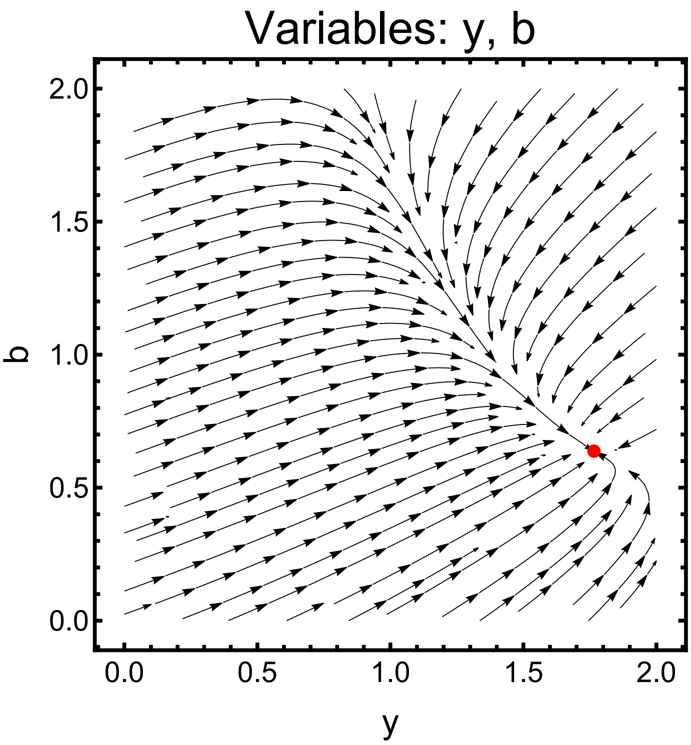
In[41]:=
```
RM2c5 = Solve[{z^(1/q) - y*b*a^g - y*a^d == 0, x*a^f - y*b*a^g == 0}, {y, b}]
```

Steady State at given values

In[42]:=
```
solRM2c5 = RM2c5[[1]] /. {x → x0, z → z0, a → a0};
ysolc5 = (Part[solRM2c5, 1] /. Rule → List)[[2]];
bsolc5 = (Part[solRM2c5, 2] /. Rule → List)[[2]];
```

In[45]:=
```
Show[StreamPlot[{z0^(1/q) - y*b*a0^g - y*a0^d, x0*a0^f - y*b*a0^g},
  {y, 0, 2}, {b, 0, 2}, Frame → True, FrameStyle → Directive[Black, 15],
  FrameLabel → {Style["y", 18], Style["b", 18]},
  FrameTicks → Automatic, FrameTicksStyle → Directive[Thick, Black, 15],
  LabelStyle → Directive[Black], PlotLabel → Style["Variables: y, b", Large],
  StreamColorFunction → None, StreamStyle → Black],
 Graphics[{PointSize[Large], Red, Point[{ysolc5, bsolc5}]}],
 FrameStyle → {Thick, Directive[Thick, Black]}]
```

Out[45]=



Steady States Case 7

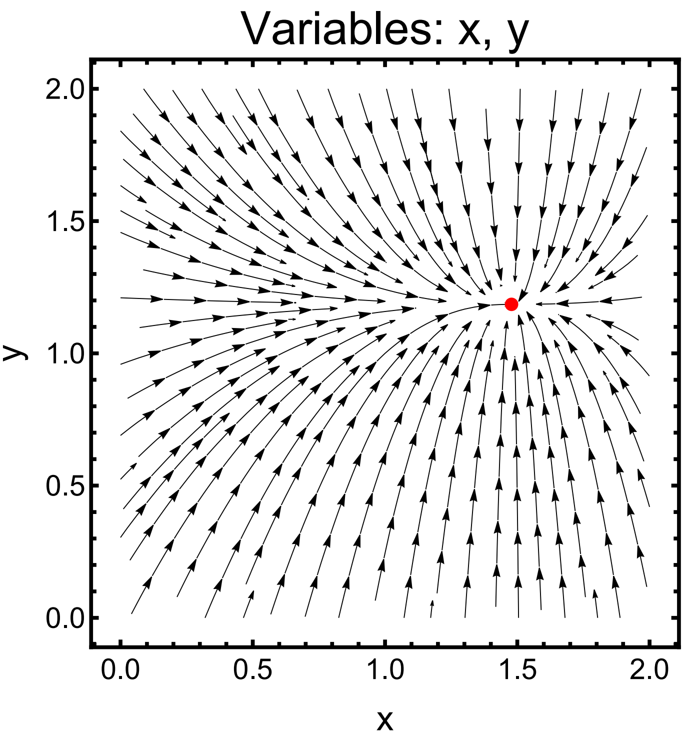
In[46]:=
```
RM2c7 = Solve[{z^(1/p) - x*a^f == 0, z^(1/q) - y*b*a^g - y*a^d == 0}, {x, y}]
```

1.1 Steady State at given values

In[47]:=
```
solRM2c7 = RM2c7[[1]] /. {z → z0, a → a0, b → b0};
xsolc7 = (Part[solRM2c7, 1] /. Rule → List)[[2]];
ysolc7 = (Part[solRM2c7, 2] /. Rule → List)[[2]];
```

In[50]:=
```
Show[StreamPlot[{z0^(1/p) - x*a0^f, z0^(1/q) - y*b0*a0^g - y*a0^d},
  {x, 0, 2}, {y, 0, 2}, Frame → True, FrameStyle → Directive[Black, 15],
  FrameLabel → {Style["x", 18], Style["y", 18]},
  FrameTicks → Automatic, FrameTicksStyle → Directive[Thick, Black, 15],
  LabelStyle → Directive[Black], PlotLabel → Style["Variables: x, y", Large],
  StreamColorFunction → None, StreamStyle → Black],
 Graphics[{PointSize[Large], Red, Point[{xsolc7, ysolc7}]}],
 FrameStyle → {Thick, Directive[Thick, Black]}]
```

Out[50]=



Type of critical point

Calculate the Jacobian

In[51]:=
```
Jc7 = D[{z^(1/p) - x*a^f, z^(1/q) - y*b*a^g - y*a^d}, {{x, y}}]
```

Calculate the Jacobian for the found steady states

In[52]:=
```
Jc71 = Jc7 /. RM2c7[[1]]
```

Eigen value equation (expansion of the characteristic equation)

In[53]:=
```
eve7 = Solve[l^2 - l*Tr[Jc71] + Det[Jc71] == 0, l]
```

Analyse the sign of Tr[J], Det[J] and Tr[J]^2 - 4 Det[J]

In[54]:=
```
Trc7 = Tr[Jc71]
Dc7 = Det[Jc71]
Sc7 = Tr[Jc71]^2 - 4 Det[Jc71]
Simplify[Sc7]
```

Steady States Case 8

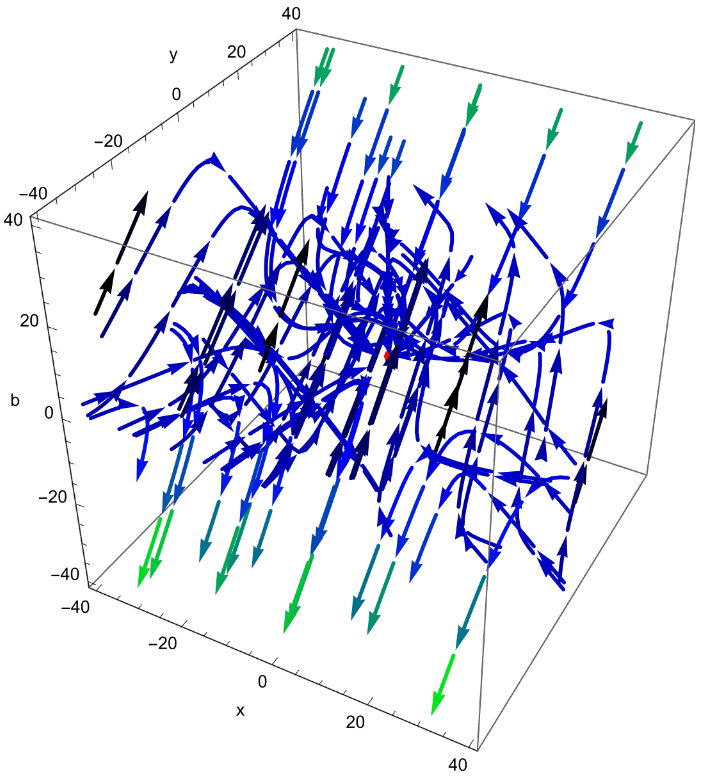
In[58]:=
```
RM2c8 = Solve[{z^(1/p) - x*a^f == 0,
   z^(1/q) - y*b*a^g - y*a^d == 0, x*a^f - y*b*a^g == 0}, {x, y, b}]
```

Steady State at given values

In[59]:=
```
solRM2c8 = RM2c8[[1]] /. {z → z0, a → a0};
xsolc8 = (Part[solRM2c8, 1] /. Rule → List)[[2]];
ysolc8 = (Part[solRM2c8, 2] /. Rule → List)[[2]];
bsolc8 = (Part[solRM2c8, 3] /. Rule → List)[[2]];
```

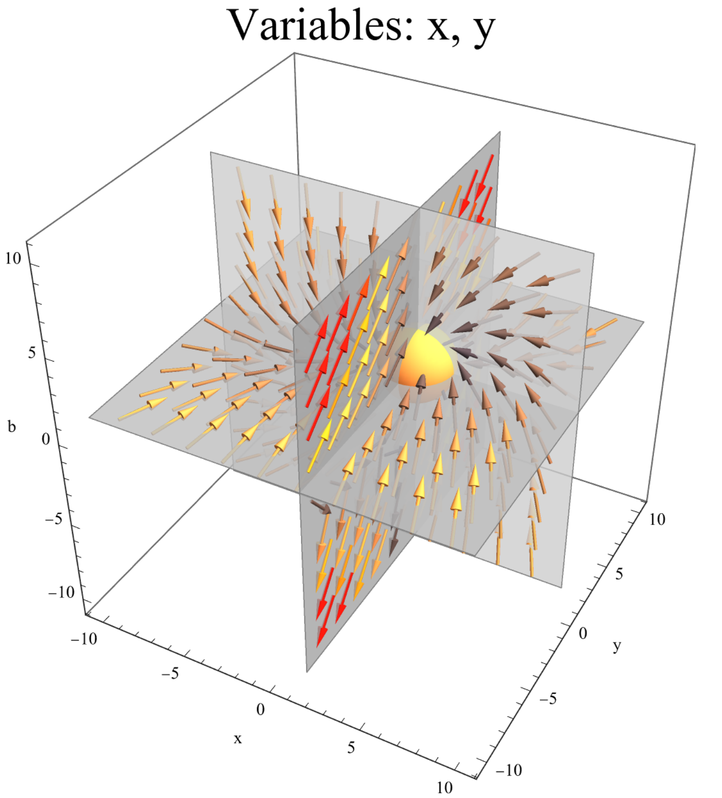
In[63]:=
```
Show[StreamPlot3D[
  {z0^(1/p) - x*a0^f, z0^(1/q) - y*b*a0^g - y*a0^d, x*a0^f - y*b*a0^g},
  {x, -40, 40}, {y, -40, 40}, {b, -40, 40}, StreamMarkers → "Arrow",
  StreamColorFunction → (Blend[{Green, Blue, Black}, #5] &), StreamPoints → 60,
  StreamStyle → Thickness[2], AxesLabel → {"x", "y", "b"}],
 Graphics3D[{PointSize[Large], Red, Point[{xsolc8, ysolc8, bsolc8}]}]]
```

Out[63]=



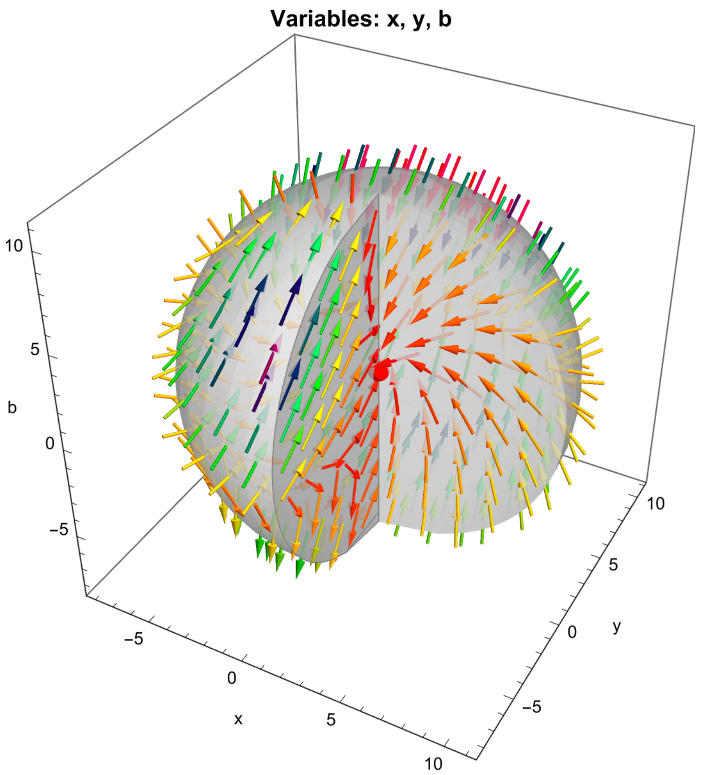
In[64]:=
```
Show[SliceVectorPlot3D[
   {z0^(1/p) - x*a0^f, z0^(1/q) - y*b*a0^g - y*a0^d, x*a0^f - y*b*a0^g},
   {x == xsolc8, y == ysolc8, b == bsolc8}, {x, -10, 10}, {y, -10, 10},
   {b, -10, 10}, PlotTheme → "Scientific", VectorAspectRatio → 0.2,
   VectorColorFunction → "TemperatureMap", AxesLabel → {"x", "y", "b"},
   PlotLabel → Style["Variables: x, y", Large]],
  Graphics3D[Sphere[{xsolc8, ysolc8, bsolc8}, 2]]]
```

Out[64]=



In[65]:=
```
Show[SliceVectorPlot3D[
  {z0^(1/p) - x*a0^f, z0^(1/q) - y*b*a0^g - y*a0^d, x*a0^f - y*b*a0^g},
  "CenterCutSphere", {x, xsolc7 - 9, xsolc7 + 9}, {y, ysolc8 - 9, ysolc8 + 9},
  {b, bsolc8 - 9, bsolc8 + 9}, PlotTheme → "Scientific",
  VectorAspectRatio → 0.2, VectorColorFunction → Hue,
  AxesLabel → {"x", "y", "b"}, LabelStyle → Directive[Black],
  PlotLabel → Style["Variables: x, y, b", Medium, Bold]],
 Graphics3D[{Red, Sphere[{xsolc8 , ysolc8, bsolc8}, 0.5]}]]
```

Out[65]=



Type of critical point

Calculate the Jacobian

In[66]:=
```
Jc8 =
 D[{z^(1/p) - x*a^f, z^(1/q) - y*b*a^g - y*a^d, x*a^f - y*b*a^g}, {{x, y, b}}]
```

Calculate the Jacobian for the found steady states

In[67]:=
```
Jc81 = Jc8 /. RM2c8[[1]]
```

Eigen value equation (expansion of the characteristic equation)

In[68]:=
```
eve8 = Solve[l^2 - l*Tr[Jc81] + Det[Jc81] == 0, l]
```

Analyse the sign of Tr[J], Det[J] and Tr[J]^2 - 4 Det[J]

In[69]:=
```
Trc8 = Tr[Jc81]
Dc8 = Det[Jc81]
Sc8 = Tr[Jc81]^2 - 4 Det[Jc81]
Simplify[Sc8]
```

Cases from Table A4 (Appendix D)

RM1

Case 9 : x' = 0, var y, a

Case 10 : y' = 0, var x, a

RM2s

Case 11 : x' = b'= 0, var y, a

Case 12 : y' = b'= 0, var x, a

RM2

Case 13 : x' = b'= 0, var y, a

Case 14 : y' = b'= 0, var x, a

Case 15 : x'= 0, var y, a, b

Case 16 : y' = 0, var x, a, b

Case 17 : a'= 0, var x, y, b

1. Steady states :

```
In[73]:= RM1c9 = Solve[{z^(1/q) - y*a^d == 0, -x*a^c - y*a^d == 0}, {y, a}]

In[74]:= RM1c10 = Solve[{z^(1/p) - x*a^c == 0, -x*a^c - y*a^d == 0}, {x, a}]

In[75]:= RM2sc11 = Solve[{z^(1/q) - y*b*a^g == 0, -x*a^f - y*b*a^g == 0}, {y, a}]

In[76]:= RM2sc12 = Solve[{z^(1/p) - x*a^f == 0, -x*a^f - y*b*a^g == 0}, {x, a}]

In[77]:= RM2c13 =
          Solve[{z^(1/q) - y*b*a^g - y*a^d == 0, -x*a^f - y*b*a^g - y*a^d == 0}, {y, a}]

In[78]:= RM2c14 = Solve[{z^(1/p) - x*a^f == 0, -x*a^f - y*b*a^g - y*a^d == 0}, {x, a}]

In[79]:= RM2c15 = Solve[{z^(1/q) - y*b*a^g - y*a^d == 0,
            -x*a^f - y*b*a^g - y*a^d == 0, x*a^f - y*b*a^g == 0}, {y, a, b}]

In[80]:= RM2c16 = Solve[{z^(1/p) - x*a^f == 0,
            -x*a^f - y*b*a^g - y*a^d == 0, x*a^f - y*b*a^g == 0}, {x, a, b}]

In[81]:= RM2c17 = Solve[{z^(1/p) - x*a^f == 0,
            z^(1/q) - y*b*a^g - y*a^d == 0, x*a^f - y*b*a^g == 0}, {x, y, b}]
```