First, Let us define the initial shape of the potential. i.e., choosing the boundaries and initialize some values for the rest of the potential. Good initialization will yield less convergence time.

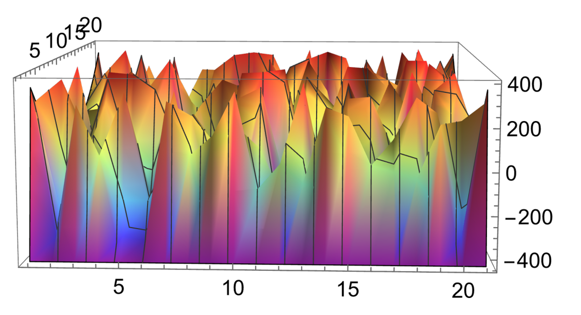
```
SeedRandom[0];
```

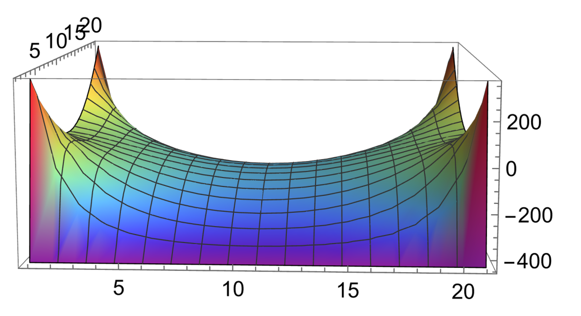
This is the template of defining the potential

```
Temp := (GridSize = 51;
  Mid = Ceiling[GridSize / 2]; (*Locating the mid-point*)
  V = ConstantArray[0, {GridSize, GridSize}]; (*Initializing V as 2-D array*)
  B1[x_] := 0;
  (*B1-4 are the boundary values. You can set them as an equation which will be evaluated from -
   GridSize to GridSize. Or you can set it to be a constant*)
  B2[x_] = 0;
  B3[x_] = 0 ;
  B4[x_] = 0;
  Table[V[[i]][[j]] = RandomInteger[{-GridSize^2, GridSize^2}], {i, 1, GridSize, 1}, {j, 1, GridSize}];
  (*Filling the 2-D array with random values*)
  List1 = Table[B1[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List2 = Table[B2[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List3 = Table[B3[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List4 = Table[B4[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  V[[ ;; , GridSize]] = List1;
  V[[ ;; , 1]] = List2;
  V[[1, ;;]] = List3;
  V[[GridSize, ;;]] = List4; (*The above were just computations to enforce the boundaries*)
  ListPlot3D[V, PlotRange → All, ColorFunction → "Rainbow"])

Saddle := (GridSize = 21;
  Mid = Ceiling[GridSize / 2];
  V = ConstantArray[0, {GridSize, GridSize}];
  B1[x_] := x x; B2[x_] = x x; B3[x_] = -21 * 21 ; B4[x_] = -21 * 21;
  Table[V[[i]][[j]] = RandomInteger[{-400, 400}], {i, 1, GridSize, 1}, {j, 1, GridSize}];
  List1 = Table[B1[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List2 = Table[B2[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List3 = Table[B3[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List4 = Table[B4[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  V[[ ;; , GridSize]] = List1;
  V[[ ;; , 1]] = List2;
  V[[1, ;;]] = List3;
  V[[GridSize, ;;]] = List4;
  ListPlot3D[V, PlotRange → All, ColorFunction → "Rainbow"])


Lincline := (GridSize = 21;
  Mid = Ceiling[GridSize / 2];
  V = ConstantArray[0, {GridSize, GridSize}];
  B1[x_] := x x; B2[x_] = -x x; B3[x_] = 21 x ; B4[x_] = 21 x;
  Table[V[[i]][[j]] = RandomInteger[{-400, 400}], {i, 1, GridSize, 1}, {j, 1, GridSize}];
  List1 = Table[B1[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List2 = Table[B2[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List3 = Table[B3[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List4 = Table[B4[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  V[[ ;; , GridSize]] = List1;
  V[[ ;; , 1]] = List2;
  V[[1, ;;]] = List3;
  V[[GridSize, ;;]] = List4;
  ListPlot3D[V, PlotRange → All, ColorFunction → "Rainbow"])

AllLin := (GridSize = 21;
  Mid = Ceiling[GridSize / 2];
  V = ConstantArray[0, {GridSize, GridSize}];
  B1[x_] := 3 x + 7; B2[x_] = -5 x + 15; B3[x_] = 7 x + 9; B4[x_] = 9 x + 21;
  Table[V[[i]][[j]] = RandomInteger[{-200, 200}], {i, 1, GridSize, 1}, {j, 1, GridSize}];
  List1 = Table[B1[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List2 = Table[B2[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List3 = Table[B3[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  List4 = Table[B4[i], {i, -GridSize, GridSize, (2*GridSize)/(GridSize - 1)}];
  V[[ ;; , GridSize]] = List1;
  V[[ ;; , 1]] = List2;
  V[[1, ;;]] = List3;
  V[[GridSize, ;;]] = List4;
  ListPlot3D[V, PlotRange → All, ColorFunction → "Rainbow"])

rRelax[s_] := For[{i = 1, j = True}, j, i++, OldV = V;
   Pause[s];
   Table[V[[i]][[j]] = N[(V[[i + 1]][[j]] + V[[i - 1]][[j]] + V[[i]][[j - 1]] + V[[i]][[j + 1]]) * 0.25], {i, 2, GridSize - 1, 1}, {j, 2, GridSize - 1, 1}];
   p = ListPlot3D[V, PlotRange → All, ColorFunction → "Rainbow"];
   If[Max[Abs[V - OldV]] ≤ tolerance, j = False;
    Print[p];
    Print["This was achieved in ", i, " iteration"];
    Print["The value of the center is equal to: ", V[[Mid]][[Mid]]];
    Print["The average value of the boundaries is: ", Mean[V[[ ;; , GridSize]] +
         V[[ ;; , 1]] +
         V[[1, ;;]] +
         V[[GridSize, ;;]]] / 4 // N]]
   ;];

Relax := For[{i = 1, j = True}, j, i++, OldV = V;
   Table[V[[i]][[j]] = N[(V[[i + 1]][[j]] + V[[i - 1]][[j]] + V[[i]][[j - 1]] + V[[i]][[j + 1]]) * 0.25], {i, 2, GridSize - 1, 1}, {j, 2, GridSize - 1, 1}];
   p = ListPlot3D[V, PlotRange → All, ColorFunction → "Rainbow"];
   If[Max[Abs[V - OldV]] ≤ tolerance, j = False;
    Print[p];
    Print["This was achieved in ", i, " iteration"];
    Print["The value of the center is equal to: ", V[[Mid]][[Mid]]];
    Print["The average value of the boundaries is: ", Mean[V[[ ;; , GridSize]] +
         V[[ ;; , 1]] +
         V[[1, ;;]] +
         V[[GridSize, ;;]]] / 4 // N]]
   ;];
```

Now we will visualize some potentials before and after relaxation; Set the value of the tolerance, the smaller the preciser but at the cost of more iterations:

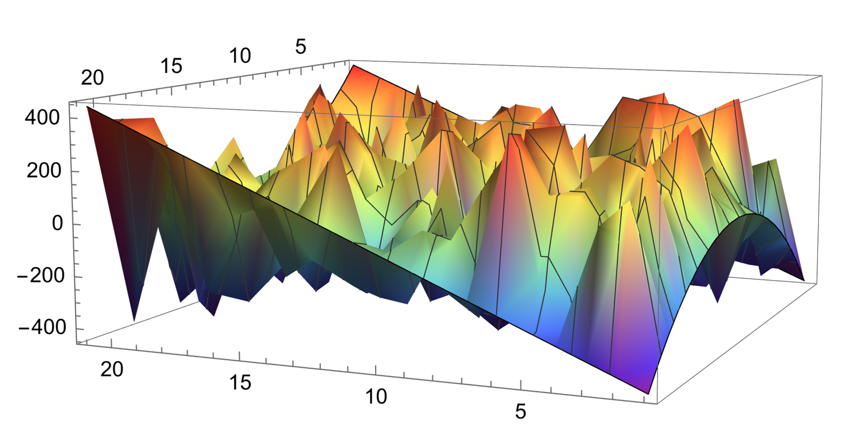
```
tolerance = 1 * 10^-4; (*The condition of convergence is Max(|V-OldV|≤tolerance)*)

Saddle
Relax
```

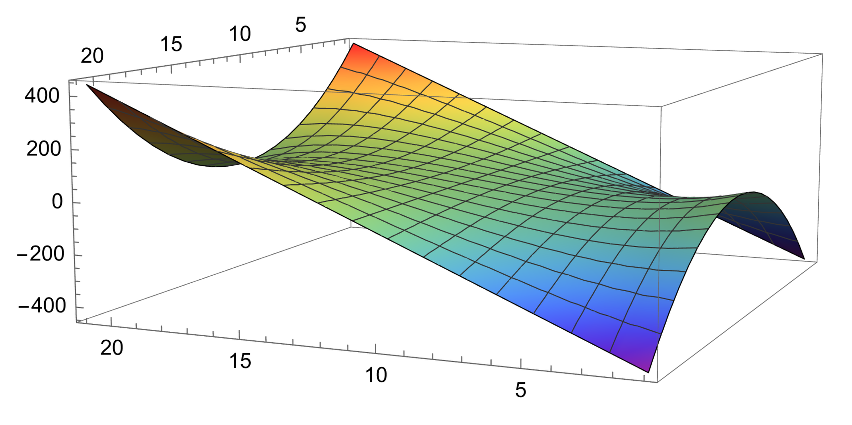




```
This was achieved in 456 iteration
The value of the center is equal to: -181.594
The average value of the boundaries is: -181.65
```

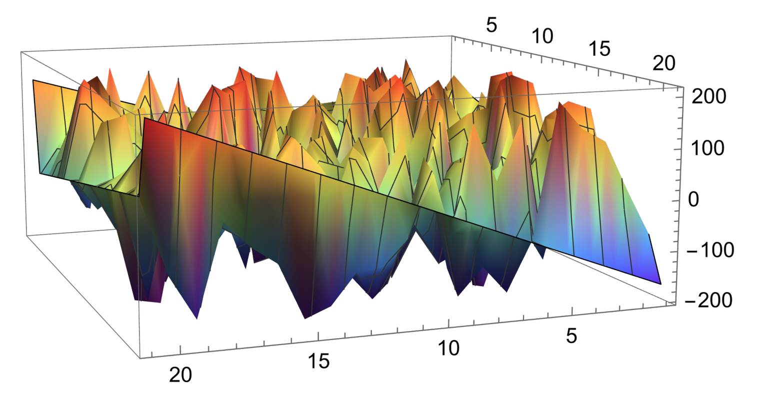
```
Lincline
Relax
```

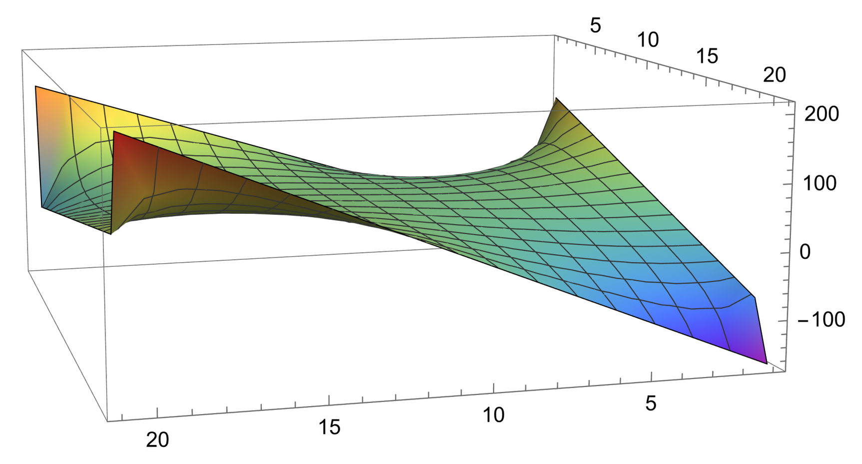




```
This was achieved in 291 iteration
The value of the center is equal to: -0.00393734
The average value of the boundaries is: 0.
```

```
AllLin
Relax
```





```
This was achieved in 342 iteration
The value of the center is equal to: 12.996
The average value of the boundaries is: 13.1905
```

Here you can see the relaxation, change the argument of rRelax to make it slower

```
Saddle; Monitor[rRelax[0.8];, p]
Lincline; Monitor[rRelax[0.8];, p]
AllLin; Monitor[rRelax[0.8];, p]
```