

# Random Walk

*File: RandomWalk.mw*

— Show the path and distance of last position from first

```
> #determine points of path
randomize():
x := 0:
x0 := 0:
y := 0:
y0 := 0:
n := 25: # number of steps

rand01 := rand(0..1):

lst := [[0, 0]]:
for i from 1 to n do
  if (rand01() = 0) then
    x := x + 1
  else
    x := x - 1
  end if:
  if (rand01() = 0) then
    y := y + 1
  else
    y := y - 1
  end if:
  lst := [op(lst), [x, y]]
end do:

> # display points in blue
with(plots):
lp1 := listplot(lst, style = point, color = blue,
  axes= none):

# plot lines connecting points
lp2 := listplot(lst, style = line, color = black,
  axes= none):

# display first and last points in larger in blue
lastPoint := listplot([[0,0], [x, y]],
  style = point, symbol = circle, symbolsize = 15,
  color = blue, axes= none):

# show points and lines of path
display([lp1, lp2, lastPoint]);

# distance between first and last points
evalf(sqrt((x - x0)^2 + (y - y0)^2));
```

— Average over many runs to find average distance between first and last points

```

> # set up
randomize():
numTests := 100: # number of runs
x0 := 0:
y0 := 0:
n := 25: # number of steps

rand01 := rand(0..1):
sumDist := 0: # sum of distances

# generate numTests number of random paths
for j from 1 to numTests do
  x := x0:
  y := y0:

  for i from 1 to n do
    if (rand01() = 0) then
      x := x + 1
    else
      x := x - 1
    end if:
    if (rand01() = 0) then
      y := y + 1
    else
      y := y - 1
    end if:
    lst := [op(lst), [x, y]]
  end do:

  # distance between first and last points
  sumDist := sumDist +
    evalf(sqrt((x - x0)^2 + (y - y0)^2)):
end do:

# average distance between first and last points
evalf(sumDist/numTests);

```

**[-] Generate list of average distances in  $n$  steps, where  $n$  varies from 1 to 50**

```

> # set up
randomize():
numTests := 100: # number of runs
x0 := 0:
y0 := 0:
n := 'n': # clear number of steps
rand01 := rand(0..1):

listDist := []: # list of distances

# travel n steps for n from 1 to 50
for n from 1 to 50 do
  sumDist := 0: # sum of distances

  # generate numTests random paths of n steps each
  for j from 1 to numTests do

```

```

x := x0:
y := y0:

for i from 1 to n do
  if (rand01() = 0) then
    x := x + 1
  else
    x := x - 1
  end if:
  if (rand01() = 0) then
    y := y + 1
  else
    y := y - 1
  end if:
end do:

# distance between first and last points
sumDist := sumDist +
  evalf(sqrt((x - x0)^2 + (y - y0)^2)):
end do:

# append average distance traveled in n steps
listDist := [op(listDist),
  evalf(sumDist/numTests)]:
end do:

# return list of distances
listDist;

```

– Determine the relationship of distance to  $n$ , the number of steps by discovering an empirical model to fit the data.

```

> plots[listplot](listDist,
  style = point,
  labels = ["n", "avg. dist."]);

```

