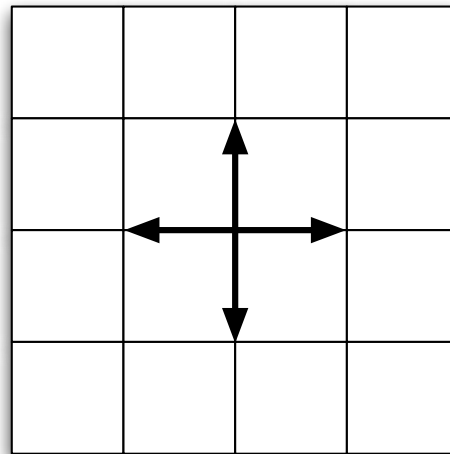


モンテカルロ法

- 乱数を用いてシミュレーションや数値計算を行う手法の総称
- 物理学や生物学などのシミュレーションに良く用いられる
- 具体例：コイン投げ、ランダムウォーク（乱歩）、など、確率論的な事象の変化をアルゴリズムとして記述して実行する

格子上のランダムウォーク

- 2次元格子空間を考える。各個体は格子の上に存在し、単位時間内に隣接する4つの格子のいずれかへ等しい確率 $1/4$ で移動する。
- 初期分布として原点に N 個体存在する状態を考える。時刻 t での個体の空間分布はどのようなものか？ ($t = 0, 1, 2, 3, \dots$)
- また、時刻 t と原点から最も離れた個体の距離との関係は？



格子上のランダムウォーク

- N 個体の位置を 2次元配列で表現するランダムウォークに取り組む
- 初期状態として全ての個体は原点に位置するとする
- 以下を繰り返す (時刻のループ)
 - 次のアルゴリズムに従って、 N 個体の座標を変化させる

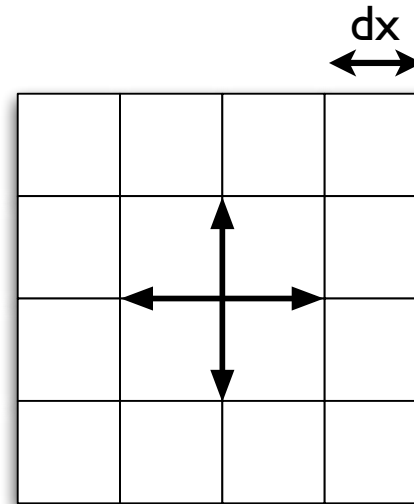
各個体の x, y 座標は

確率 $1/4$ で $x += dx$

確率 $1/4$ で $x -= dx$

確率 $1/4$ で $y += dx$

確率 $1/4$ で $y -= dx$



プログラム骨格

```
#define N 100
#define DX 1.0

double x[N], y[N] // 個体の座標

initialize(); // 初期化：すべての個体は原点にある

for(step=0; step<STEP; step++){
    move_indivs(); // 個体を移動させる関数
    write_data(); // 個体の位置をファイルに書き出す
}

void move_indivs()
{
    ....
}

void write_data()
{
    ....
}
```

プログラム骨格

```
void move_indivs()
{
    int i;
    double rv;

    for(i=0; i<N; i++){

        rv = genrand_real2();

        if( rv < 0.25 )    // rv の値に応じてx, y座標を更新
            x[i] += DX;
        else if( rv < 0.50 )
            x[i] -= DX;
        else if( rv < 0.75 )
            y[i] += DX;
        else
            y[i] -= DX;
    }
}

void write_data()
{
    int i;

    for(i=0; i<N; i++)
        fprintf(fp, "%f %f\n", x[i], y[i]);

    fprintf(fp, "\n\n"); // gnuplot でアニメ表示のため改行を2回書き出す
}
```

計算結果の表示

anime.txt

を作成

```

if (i == 0) set size square
if (i == 0) set size 1.0
if (i == 0) set xrange[-30:30]    ← x座標の範囲を設定
if (i == 0) set yrange[-30:30]   ← y座標の範囲を設定
print i
plot "data-rw" index i title "random walk" ps 2 ←
i=i+1                               読み込むデータ
pause 0.1                             ファイルを指定
if (i<50) reread

```

← 最後まで読み終わったらやり直し

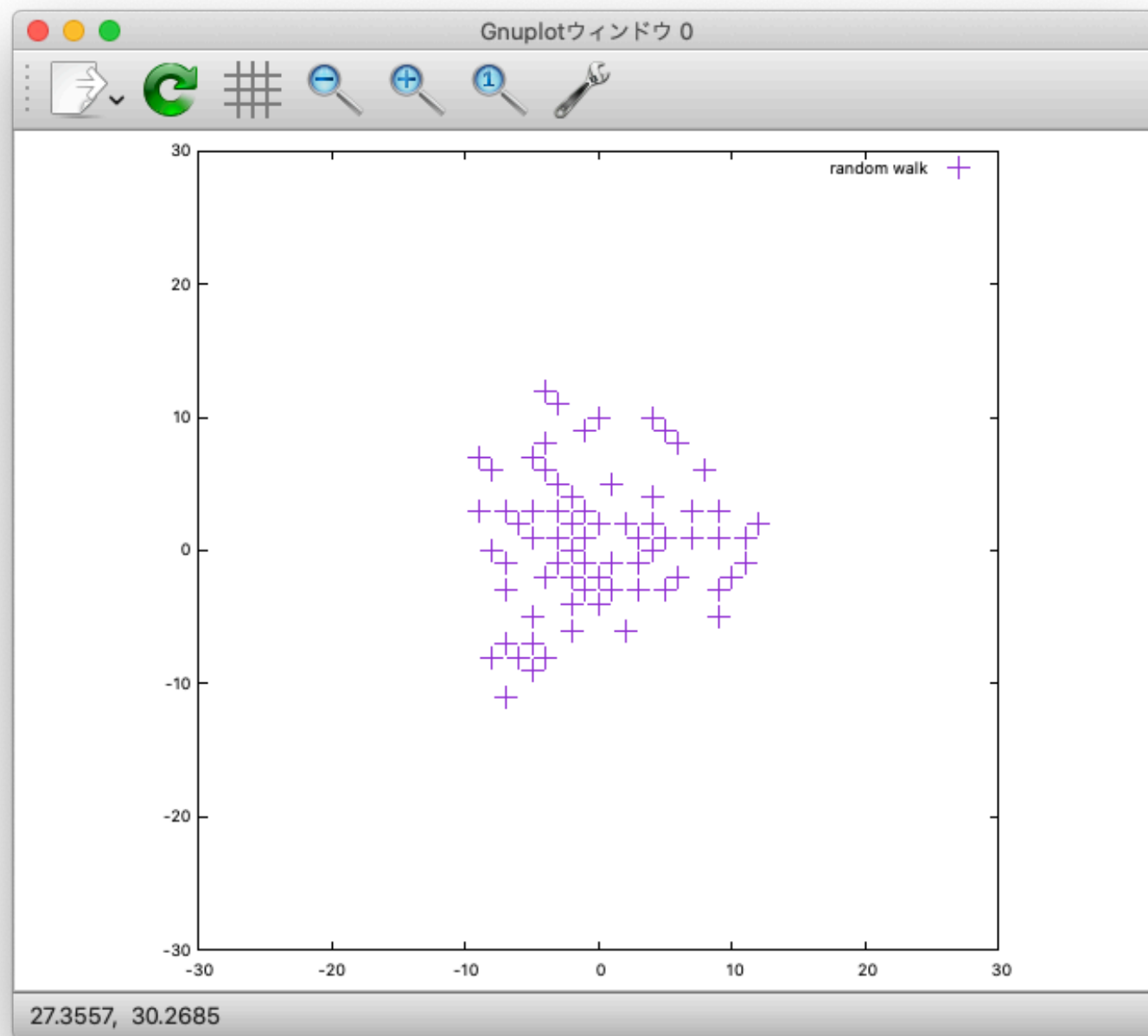
gnuplot でアニメ（紙芝居）を表示

```

gnuplot> i=0
gnuplot> load "anime.txt"

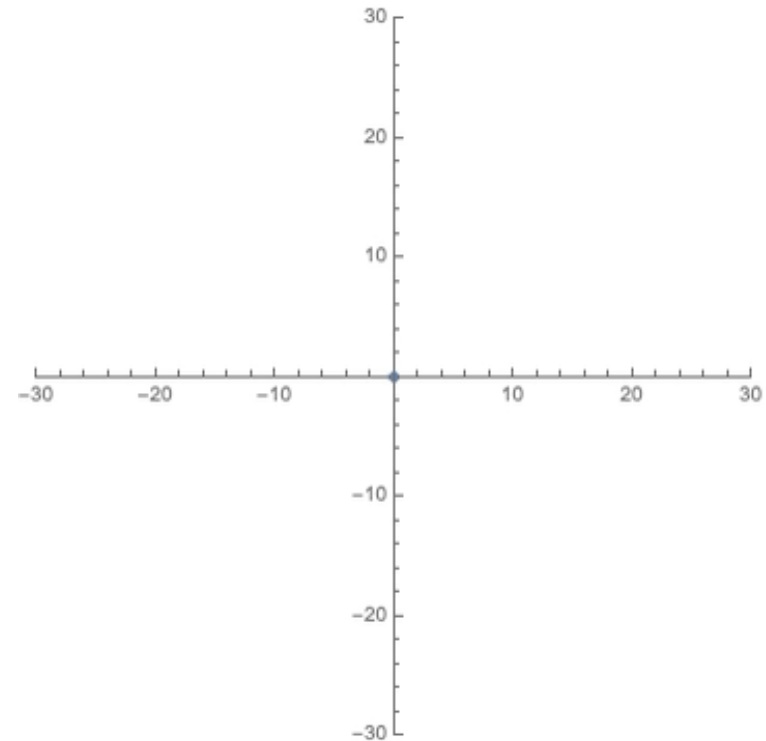
```

計算結果の表示



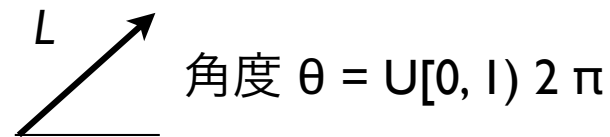
2次元格子上の乱歩

```
data = ReadList[ "data", Real, RecordLists -> True ];  
data2 = Map[ Partition[#, 2] &, data];  
Length[data2]  
{ Max[data2], Min[data2] }  
  
Do[  
  ListPlot[ data2[[i]], PlotRange -> {{-40, 40}, {-40, 40}}, AspectRatio -> 1 ]//Print, {i, 1,  
Length[data2]}  
]
```



問題 7

- 個体の位置は格子状ではなく、連続空間上にあるとする (x, y 座標が実数)
- 単位時間後の個体の移動に関して**適当なルール**を設定し、 N 個体の移動分散の様子を調べよ。ルールとしては例えば下記が考えられる
 - ルール 1: 一定の距離 L だけ移動するが、方角は任意の方向に等しい確率で移動
 - ルール 2: 移動距離は一定 L だが、方向は N 個体の重心へ向かう、など



自分で設定したルールの下で、空間分布の時間変化および時刻 t と原点から最も離れた個体の距離との関係を調べよ