関数であそぼう

関数の遊び方を知っている利点

- 保守性
  - 再利用性
  - モジュラリティ

- 楽しい
  - 高尚なことをやっている感じ

高階関数(関数を引数として受け取る関数)

主なもの: map, filter

map (リストの要素すべてに行いたい処理，リスト)

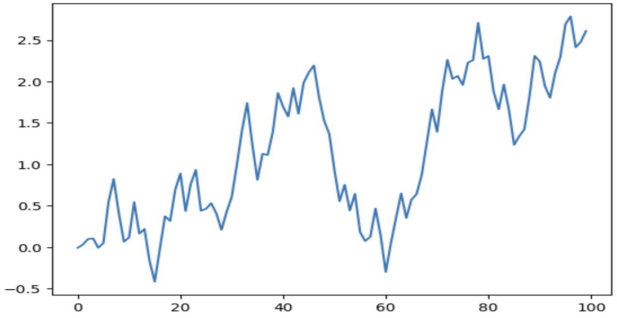
例:

map((*2), [1,2,3]) = [2,4,6]

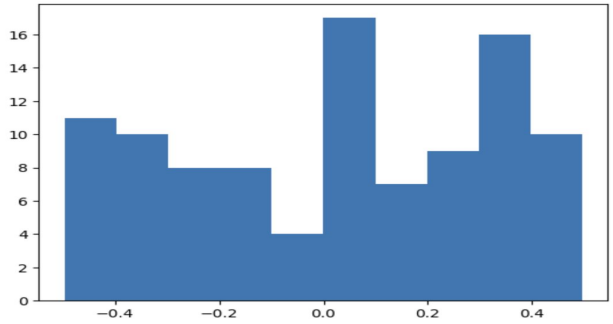
filter(odd, [1,2,3,4,5,6,7]) = [1,3,5,7]

高階関数の自作例：折れ線グラフ、ヒストグラムのタイル表示

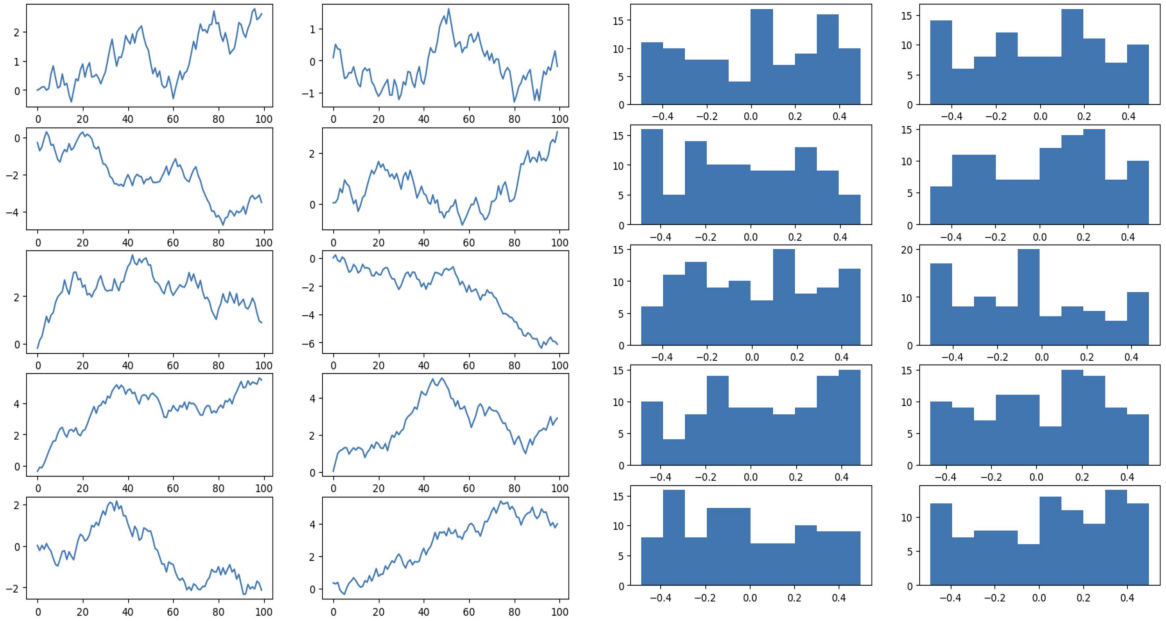
cumsumbar(data: List[float])



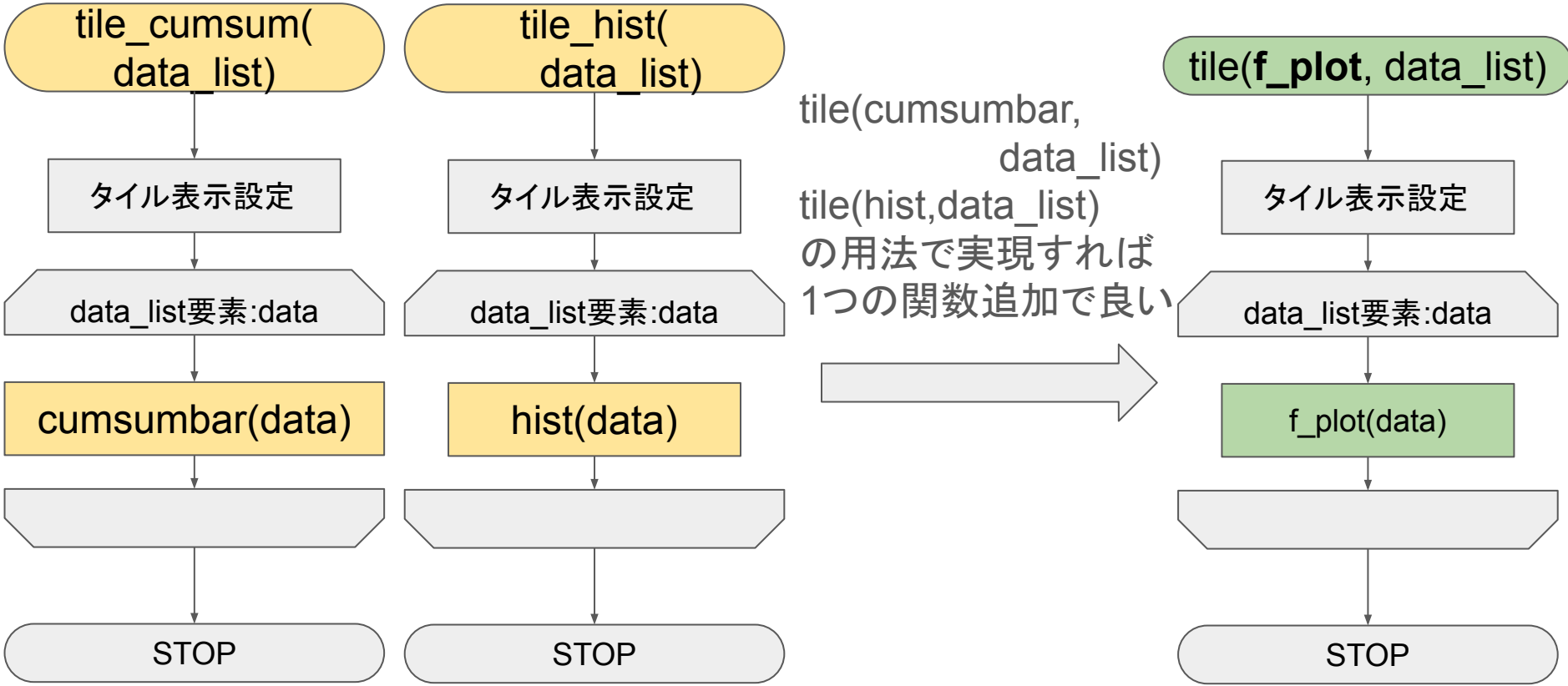
hist(data: List[float])



複数のデータ列に対して
タイル状にプロットしたい



高階関数の自作例：折れ線グラフ、ヒストグラムのタイル表示



関数合成：部署内の全員が汚染地域外に居住か？

func_1の出力をfunc_2に入力すること

$$(func\_2 \cdot func\_1)(x) = func\_2(func\_1(x))$$

利点：中継用の変数が不要

入力：["XX市YY町00-00", "AA市BB町11-11", ...]

f_comp = func_閾値以上か(距離→True/False)

・func_汚染地点からの距離(緯度経度→距離)

・func_住所からの緯度経度変換(住所→緯度経度)

判定：all(map(f_comp, data))

部分適用: partial:一部引数のみ決定

func_汚染地点からの距離(緯度経度→距離)

ローカル定義した緯度経度とのユークリッド距離か、、、
汎用的に出来ないか？

func_dist(緯度経度A,緯度経度B→距離)

func_汚染地点からの距離(緯度経度→距離)

=partial(func_dist, 緯度経度A ← 汚染地点) (緯度経度→距離)
=lambda b: func_dist(a,b)

楽しみ方を知った。次は何を？

- Haskellなどの関数型言語がお勧め。
- より高度な概念であるファンクタ(関手)やモナドの学習