The algorithmic beauty of plantsを基に、L-Systemを実装した。

# L-System

1.7章（P.28）のStochastic L-Systemを実装した。

動作確認したルールは、Figure 1.24、Figure 1.25、Figure 1.27。

# Parametric L-System

1.10章（p.40）のParametric L-Systemを実装した。ただし、conditionについては、まだ動作確認していない。

動作確認したルールは、Figure 2.6、Figure 2.7、Figure 2.8（ただし、branchのbendingは実装していない）。

L-Systemによる木の生成では、いろいろ制限があることが分かった。例えば、再帰数で再帰する回数を制限する方法だと、木の下の方の枝別れの多さに対して、木の上の方の枝分かれが極端に少なく見える。また、branchのbendingについては、例えばp.58のTropismで説明しているように、ルールとは別に、木の角度を変えるという実装が必要になる。結局、ルールでは表現しきれないということだ。だったら、そもそもL-Systemにこだわる必要はないのでは？

# Inverse L-System

自分のInverse研究用に、いろいろカスタマイズ。

# DerivationTree

事前にサンプルを生成し、Derivation state treeを作成する。ターゲットに近いサンプルからスタートし、greedyにinverse生成する。

# GreedyInversePipe

いろいろ気づいたこと。

## Indicatorの予測

次のどっちのルールを適用するか？あるいは、どのパラメータ値を選択するかは、その選択によるIndicatorをestimateして決定する。Indicatorをestimateするために、そこから先に何ステップかmodelをランダムに生成してみて予測する。このステップ数は、小さすぎると十分な予測ができない。例えば、下図のように、数字「５」から「１」へジャンプするためには、一時的にコストが上昇するような選択をする必要があるため、ステップ数が小さいと、コストが高いために選択されない。その結果、数字「１」へジャンプできない。いわば、極値に陥るようなイメージだ。一方、ステップ数が多すぎると、ランダムな生成では最適な選択がされないため、逆に良い予測にならなくなる。いわば、発散してしまうようなイメージだ。ものすごい数のサンプリングを行わない限り、良い予測にならない。

