Optimization by TCNN in discrete and continuous domains

* HNN, TCNN, multimodal function optimization
* TCNN
  + Chaotic (bifurcation + lapunov)
  + Discreet (equations)
* Compare with GA, swarm, Monte Carlo, …
* TCNN analog
  + Chaotic functions
  + Equations
  + Simulink schema

<https://en.wikipedia.org/wiki/Differential_evolution>

<https://en.wikipedia.org/wiki/Memetic_algorithm>

<https://en.wikipedia.org/wiki/Graduated_optimization>

https://en.wikipedia.org/wiki/Global\_optimization

1. Introduction
2. TCNN
   1. Chaotic dynamic
   2. Compare
3. TCNN analog
   1. Chaotic analog function
   2. Experimental results
   3. Analog schema (Simulink)
4. Conclusions
5. Introduction

В данной статье представлено сравнение TCNN с различными алгоритмами для решения задач поиска глобального минимума. В статье так же рассматривается переход от дискретной модели TCNN к аналоговой, и результаты моделирования аналогового представления.

1. Описание TCNN

TCNN описанная и предложенная [] основывается на сети хопфилда и методе отжига.

TCNN описывается следующей системой уравнений

где

xi – выход нейрона

yi – внутреннее состояние нейрона

wij – весовой коэффициент между нейронами

Ii – смещение нейрона

zi – коэффициент обратной связи

Отличие от сети хопфилда заключается в том, что в начальный момент, в сети происходит хаотический поиск, который обуславливается хаотической функцией () налог температуры в методе отжига. Со временем влияние хаоса на систему уменьшается, что приводит к более выраженной динамики сети хопфилда, которая основана на градиентном методе.

Т.к. в итоге сеть сойдётся к сети хопфилда, то она гарантированно найдёт локальный минимум, но при введении начальной хаотической динамики сеть будет пытаться выйти из локального минимума.

Моделирование:

Динамика одного нейрона:

Тестовый пример поиска глобального минимума:

Аналоговое представление сети:

В статье TCNN есть описание сети в аналоговом виде, но представленная ими хаотическая функция не будет хаотической если перейти в аналоговый спектр. Поэтому для реализации аналоговой модели TCNN необходимо описать и аналоговую хаотическую функцию.

Для аппаратной поддержки возьмём функцию, которая может быть легко представлена с помощью аналоговых компонент.

Хаотическая функция:

Описание, фазовый портрет

Аналоговая математическая модель для TCNN:

Моделирование:

Динамика одного нейрона:

???

Тестовый пример поиска глобального минимума:

Сравнение с различными алгоритмами поиска глобального минимума:

- GA

- Ant Colony

- Simulating Annealing

- Monte Carlo

Пример использования аналоговой модели: ???

Выводы

В данной статье было проведено сравнение различных методов поиска глобального оптимума, среди которых TCNN отличается как метод, который может быть реализован в аналоговом представлении.

---------------- TOTAL RESULT -------------------

result: 100/50

fails: 76

---------------- TOTAL RESULT -------------------

result: 100/0

fails: 0

subprocess.check\_call([WIN\_BIN\_OPT\_FUNC

, "--steps", "8000"

, "--step\_len", "0.001"

, "--chaotic\_step\_len", "0.15"

, "--alpha", "0.05"

, "--chaotic\_coeff", "20000" #800

, "--chaotic\_reduce", "0.99993"

, "--function", "5"

, "--dim", "30"

, "--init\_cond", "0, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8, -1.8"] )

---------------- TOTAL RESULT -------------------

result: 10/0

fails: 0

subprocess.check\_call([WIN\_BIN\_OPT\_FUNC

, "--steps", "15000"

, "--step\_len", "0.001"

, "--chaotic\_step\_len", "0.15"

, "--alpha", "0.05"

, "--chaotic\_coeff", "20000" #800

, "--chaotic\_reduce", "0.5"

, "--function", "5"

, "--dim", "30"

, "--init\_cond", "0, 2.8, 3.8, 1.8, 2.8, 3.8, -1.2, -3.0, -0.8, 1.9, 1.8, 2.8, 3.8, 1.8, 2.8, 3.8, -1.2, -3.0, -0.8, 1.9, 1.8, 2.8, 3.8, 1.8, 2.8, 3.8, -1.2, -3.0, -0.8, 1.9, -1.9"])

---------------- TOTAL RESULT -------------------

result: 100/0

fails: 0

subprocess.check\_call([WIN\_BIN\_OPT\_FUNC

, "--steps", "15000"

, "--step\_len", "0.001"

, "--chaotic\_step\_len", "0.15"

, "--alpha", "0.05"

, "--chaotic\_coeff", "20000" #800

, "--chaotic\_reduce", "0.5"

, "--function", "5"

, "--dim", str(dim)

, "--init\_cond", init\_cond])

RANDOM initial conditions [-5, 5]