

xTreLoc Demo (Mar. 11th, 2025)

実行環境

本ツールはJava環境下で実行可能であり, 付属の `run_demo.bash` を実行するとシンセティックテストが実行される. なお, 結果の描画にGnuplotを必要とする.

ディレクトリ構成

`demo` ディレクトリ内の以下のファイルが必要である. 加えて, 海岸線データ[world_10m.txt](#)を取得しておく必要がある.

```
demo/
├─ catalog_ground_truth.list
├─ config.json
├─ docx
│   └─ catalog_ground_truth.png
│   └─ catalog_syn.png
│   └─ catalog_syn_std.png
│   └─ catalog_syn_trd.png
│   └─ demo_jp.md
│   └─ demo_jp.pdf (本資料)
│   └─ kdist.png
│   └─ map_njt.png
│   └─ mat_nkt.png
│   └─ style.css
├─ map_njt.plt
├─ run_demo.bash
├─ station.tbl
├─ sumdat.py
└─ xtrelloc
```

各種モード

xTreLocは以下のコマンドにより実行される。ここでは、それを保存した `xtreloc` コマンドを実行している。

```
java -jar path/to/target/xtreloc-1.0-SNAPSHOT-jar-with-dependencies.jar <mode>
```

ここで、`mode` は以下から選択される:

- **GRD**: フォーカスドランダムサーチにより、走時差の残差を最小とするグリッドを返す。STDモードの前に実行されるべきである。
- **STD**: Station-pair Double Difference法を用いた個々のイベントについての震源決定。Fortranで書かれた `hypoEcc` (Ide, 2010; Ohta et al., 2019) のjavaへの移植版であるが、`delaz4.f` などの軽微なバグを修正済み。TRDモードの前に実行されるべきである。
- **CLS**: 空間クラスタリングによって震源のネットワークを構成し、走時差についてイベント間の差分を計算する。 `hypoDD` (Waldhauser & Ellsworth, 2000) における `ph2dt` と同等の役割であり、TRDモードの前に実行する必要がある。
- **TRD**: Guo & Zhang (2016) によるTriple Difference法を用いた相対震源再決定。
- **SYN**: 震源決定モード (GRD, STD & TRD) に直接流せるシンセティックデータを作成する。
- **MAP**: 非常に簡易的に震源分布をマッピングする。緯度経度の縮尺は正確ではない。

モードを選択しない場合、以下のhelpが表示される。

```
Usage: java -jar path/to/target/xtreloc-1.0-SNAPSHOT-jar-with-dependencies.jar <mode>
```

Modes:

GRD	- Location by grid search
STD	- Location by Station-pair DD
CLS	- Spatial clustering & create triple-diff
TRD	- Re-location by Triple Difference
SYN	- Create dat files for synthetic test
MAP	- View location results on map

データフォーマット

カタログファイル

シンセティックテストにおける正解値は `catalog_ground_truth.list` 内に記載されている. catalogファイルはスペース区切りで以下の要素を記述する:

- (String) イベント時刻: 震源決定には直接使用されない (e.g., 2000-01-01T00:00:00)
- (Double) 緯度: °Nを正とする, 10進数表記 (e.g., 39.5)
- (Double) 経度: °Eを正とする, 10進数表記 (e.g., 142.4)
- (Double) 深さ: km単位で, 下向きを正とする (e.g., 5)
- (Double) 緯度方向の決定誤差: km単位であり, 震源決定には直接使用されない
- (Double) 経度方向の決定誤差: km単位であり, 震源決定には直接使用されない
- (Double) 深さ方向の決定誤差: km単位であり, 震源決定には直接使用されない
- (Double) RMS走時差残差: 単位は秒であり, 震源決定には直接使用されない
- (String) 検測ファイルのパス: イベント時刻をyymmdd.HHMMSSフォーマットで表記したファイル名 (e.g., Jan. 1st, 2000の場合, dat/000101.000000)
- (String) イベント種別:
 - SYN: SYNモードで生成されたイベント
 - GRD: GRDモードで決定されたイベント
 - STD: STDモードで決定されたイベント
 - TRD: TRDモードで決定されたイベント
 - ERR: 震源決定中にエラーが生じたイベント (airquakeなど)
 - REF: リファレンスとして固定されるイベント (CLS, TRDモードにおいて参照される)

```
2000-01-01T03:00:00 39.5 142.7 15 0 0 0 0 dat/000101.030000 SYN
2000-01-01T04:00:00 39.6 142.4 15 0 0 0 0 dat/000101.040000 SYN
```

観測点ファイル

これらのデータについて, `station.tbl` 内の観測点からの理論走時を用いてシンセティックデータが作成される. 観測点テーブルはスペースで区切られ, 以下の要素からなる:

- (String) 観測点名
- (Double) 緯度: °Nを正とする, 10進数表記
- (Double) 経度: °Eを正とする, 10進数表記
- (Double) 深さ: 下向きを正とする, km単位. 海底地震計の場合に値は正である
- (Double) P波の観測点補正值: 理論値に足される補正值 (秒)

- (Double) S波の観測点補正值: 理論値に足される補正值 (秒)

```
ST01 39.00 142.20 1.00 0.40 0.68
ST02 39.50 142.20 1.67 0.88 1.50
```

検測ファイル

SYNモードではcatalogの震源-観測点間の理論走時を計算し, パータベーションを加えた検測ファイルを生成する. 検測ファイルは以下のフォーマットに従う:

- 1行目 (ヘッダ)
 - (Double) 緯度: °Nを正とする, 10進数表記
 - (Double) 経度: °Eを正とする, 10進数表記
 - (Double) 深さ: 下向きを正とする, km単位. 海底地震計の場合に値は正である
 - (String) モード: イベントの生成モード (SYN, GRD, STD, TRD, ERR)
- 2行目 (ヘッダ)
 - (Double) 緯度誤差: km. GRDモードの場合は0
 - (Double) 経度誤差: km. GRDモードの場合は0
 - (Double) 深さ誤差: km. GRDモードの場合は0
 - (Double) 走時残差: S波の走時差残差 (秒). GRDモードの場合は-999
- 3行目以降
 - (String) 観測点1: 観測点名
 - (String) 観測点2: 観測点名
 - (Double) 走時差: S波についての走時差. 観測点 st における到達時刻 T_{st} について, $T_{st2} - T_{st1}$ の値に対応する.
 - (Double) 検測値の信頼度: `config.json` 内の `threshold` と比較され, その検測値を使用するか判断する際に参照される. 例えば, 相互相関関数の最大値を入力することで, `threshold: 0.6` とした場合には $CC > 0.6$ の検測値のみを用いて震源を決定する. SYNモードにおいては全て1が設定され, GRD, STDモードにおいては走時差の逆数の絶対値が書き込まれる. そのため, GRDモードにおいて残差が-2秒の検測値の信頼度は `0.5` であり, 次の決定ステップ (STD, TRD) において許容する残差を `threshold` で管理できる. なおSTDモードにおいては優決定問題として解くため, `threshold` を超える4つ以上の検測値がない場合にはエラーが出力される.

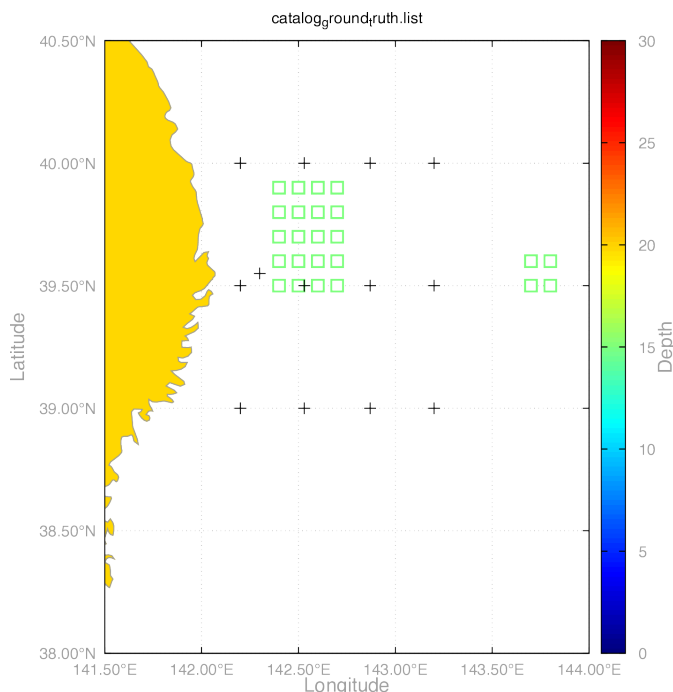
```
39.476 142.367 14.015 SYN
0.030 0.030 3.340 -999.000
ST07 ST09 -0.889 1.000
ST03 CA00 -12.975 1.000
ST02 ST09 12.280 1.000
```

シンセティックテスト

ここでは簡易的に `run_demo.bash` 内の実行内容を述べる.

まずは正解値をplotする (ただしこれは簡易的な描画であり, 必ずしも縮尺は正確でない).

```
gnuplot -e "filename='catalog_ground_truth.list'" map_njt.plt
```



この `catalog_ground_truth.list` 内の9列目に記載されたパス先のディレクトリを用意し, データを作成する.

```
mkdir -p dat-SYN
./xtreloc SYN
```



シンセティックデータ

パートバージョンの幅や使用する検測値の数は `SyntheticTest.java` 内で変更でき, `phsErr`, `locErr` の値は
ガウス分布の標準偏差に対応する.

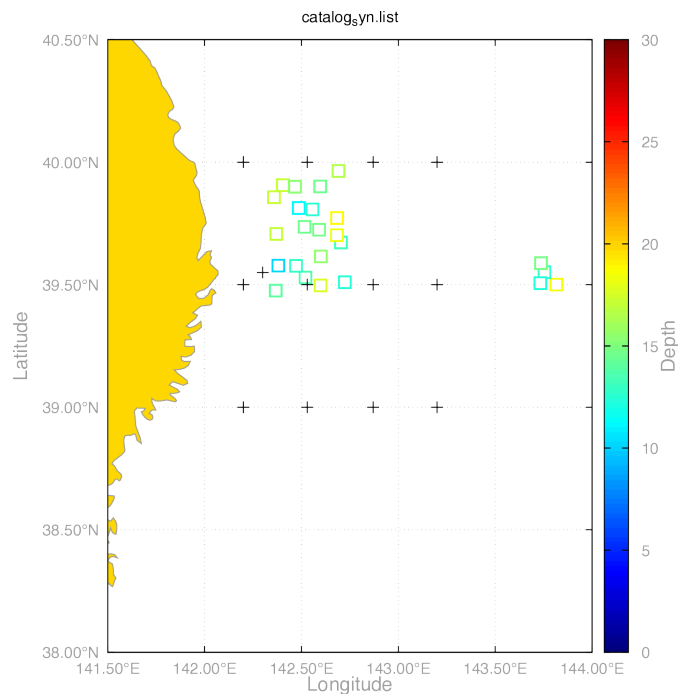
- `phsErr` : 走時差に与えるパートバージョン (default: 0.1秒)
- `locErr` : 震源に与えるパートバージョン (default: 水平 0.03度, 深さ 3km)
- `minSelectRate`, `minSelectRate` : ランダムに選定される検測の数の幅 (default: 20-40%)

```
private final double phsErr = 0.1;
private final double locErr = 0.03;
private final double minSelectRate = 0.2;
private final double maxSelectRate = 0.4;
```

結果を `sumdat.py` によりカタログ化し, Gnuplotで結果を描画する.

```
python3 sumdat.py -d dat-SYN
mv catalog.list catalog_syn.list
gnuplot -e "filename='catalog_syn.list'" map_njt.plt
```

パーティションが加わった分布は `catalog_syn.pdf` として出力される.



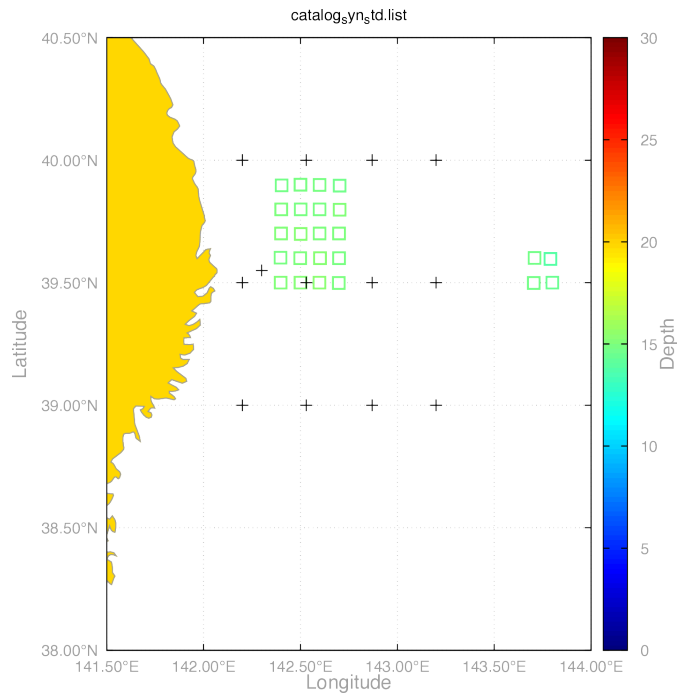
このデータに対してSTDモードで震源を決定すると, 結果は `dat-SYN_STD` に出力される.

```
./xtreloc STD
```

同様にカタログ化・描画を行う.

```
python3 sumdat.py -d dat-SYN_STD
mv catalog.list catalog_syn_std.list
gnuplot -e "filename='catalog_syn_std.list'" map_njt.plt
```

観測点から外れた震源 (143.50°E-144°E) を除き, 格子状の分布を回復できている.



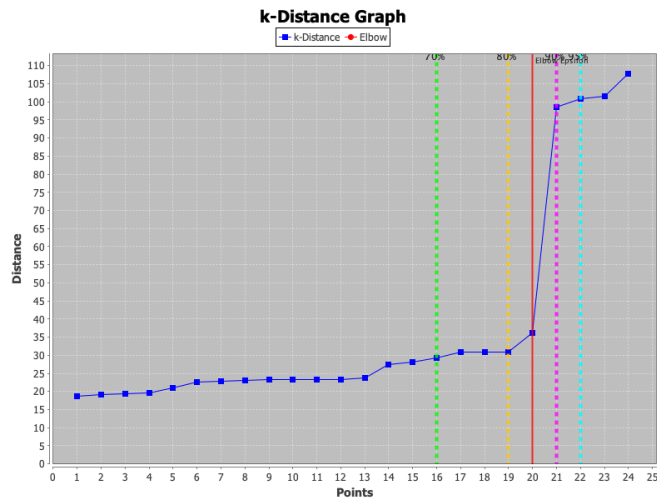
続いて, Triple Difference法についても同様のテストを行う. その準備として, クラスタリングと Triple Differenceの計算を実行する.

```
./xtreloc CLS
```

この時, `config.json` 内の `catalogFile` の11列目はクラスタ番号と認識され, ここにint型で読み込み可能な数値が記載されている場合はそのクラスタを使用する. **ただし, クラスタ番号は0から始まる連番であり, -1はクラスタに分類されないイベントに対応する.** 11列目が空の場合, `minPts` と `eps` をパラメータとするDBSCANを用いたクラスタリングが実行される. なお, `eps` が負の場合にはk-distance graphとエルボー法によって `eps` が自動で推定される.

```
"CLS": {
  "minPts": 10,
  "eps": -1,
  "catalogFile": "catalog_syn.list"
},
```

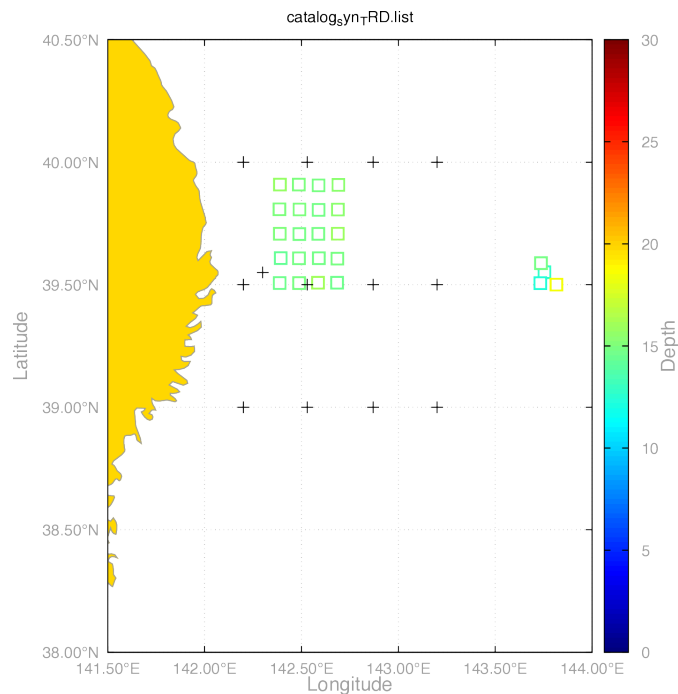
`eps` を自動で推定する場合, 以下のようなk-distance graphが出力される.



CLSモードは, `triple_diff_<cid>.csv` を出力するが, これはhypoDDにおける `dt.ct` に対応するファイルである. これを用いて相対震源決定を行う.

```
./xtreloc TRD
gnuplot -e "filename='catalog_syn_TRD.list'" map_njt.plt
```

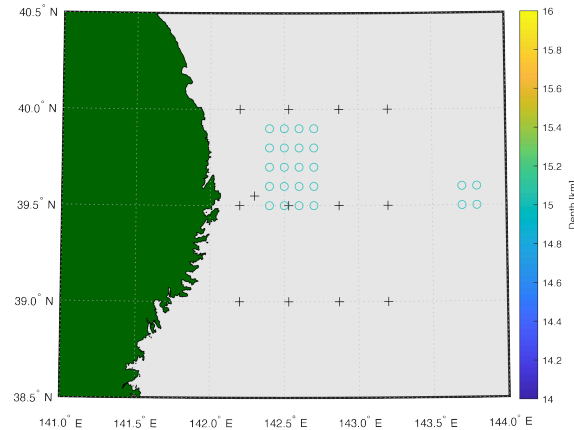
震源は以下のように決定される. 観測点から外れた, 143.50°E-144°Eに分布する4つの震源についてはクラスタ化されていない (`catalogFile` における11列目のcid=-1) ため, 入力 (`dat-SYN` 以下のファイル) をそのまま返している.



Mapping

- Matlab: `map_njt.m`

描画に用いるためのmatlabコード `map_njt.m` を添付している.



- xTreLoc (MAP mode)

xTreLocにも簡易的な描画モード (`MAP`) が存在する. 海岸線などの描画にはshapeファイルが必要で, 例えば以下から取得したものを `config.json` 内の `MAP.shpFile` に記載する.

- Natural Earth: <https://www.naturalearthdata.com/downloads/10m-physical-vectors/10m-coastline/>
- The Geospatial Information Authority of Japan (GSI):
https://www.gsi.go.jp/kankyochiri/gm_japan_e.html

例として, 以下の図は ii. に示す国土地理院の `gm-jpn-all_u_2_2/coastl_jpn.shp` を用いて描画した.

