session1

October 21, 2024

0.1 Consigne

Le travail attendu pour ces 3 semaines est à faire dans les fichiers `session1.py`, `session2.py`, `session3.py`.

Les fichiers `session{1|2|3}.py` commenceront par un commentaire avec les noms de leur auteur :

```
# prenom1 nom1
# prenom2 nom2
```

Une attention particulière sera portée à la qualité de la documentation de votre code.

Cette semaine, il s'agit de travaillr dans session1.py

[3]:
```
import session1 as project
import pandas as pd
```

1 Cultures associées et Maraîchage diversifié

L'agriculture est dominée par un modèle ou de grandes exploitations cultivent en monocultures. Pourtant, de plus en plus de données montrent que ce type d'agriculture n'est pas soutenable. En focalisant sur le maraîchage, ce sujet propose d'explorer la viabilité des fermes qui cultivent de multiples variétés de légumes.

En particulier, on étudiera dans un premier temps les données qui indiquent si la pratique qui consiste à mélanger des cultures sur une même parcelle permet d'obtenir de bons rendements. Ensuite, il s'agit de déterminer si les fermes de maraîchage diversifiés sont viable en termes de revenus et de charge de travail. Enfin, on s'interessera à la façon de réaliser ces associations dans des simulations spatialisées de la ferme.

1.1 Session 1: Analyse de données de cultures associées

On considère des associations de 2 cultures (Crop 1 et Crop 2) et on s'intéresse aux rendements de différentes couples de cultures associées (I_i = Crop_{i}_yield_intercropped) comparés à ceux des cultures seules (S_i = Crop_{i}_yield_sole). Pour une culture, on définit le ratio de terrain équivalent (LER-Land equivalent ratio) comme $LER_i = I_i/S_i$ avec $i \in \{1, 2\}$. Le LER total pour une association est $LER_{tot} = \sum_{i\in\{1,2\}} LER_i$.

On a a disposition 2 datasets qui rassemblent les rendements pour les cultures isolées et pour les cultures mélangées Ces données sont issues des publications suivantes: * Li, C., Stomph, T. J., Makowski, D., Li, H., Zhang, C., Zhang, F., & van der Werf, W. (2023). The productive performance of intercropping. Proceedings of the National Academy of Sciences, 120(2), e2201886120 *

Paut, R., Garreau, L., Ollivier, G., Sabatier, R., & Tchamitchian, M. (2024). A global dataset of experimental intercropping and agroforestry studies in horticulture. Scientific Data, 11(1), 5.

[4]:
```
d1=pd.read_csv("data/session1/dataset_PNAS_2023.csv")
d2=pd.read_csv("data/session1/dataset_natcom_2024.csv")
```

1. Ces données sont parfois incomplètes, certaines entrées ne sont pas des nombres valides. Ecrire une fonction `get_valid_indices_all_vars` qui pour une liste de variables renvoie les indices où les entrées sont des nombres valides.

[5]:
```
idxs = project.get_valid_indices_all_vars([d1['Crop_1_yield_sole'].
 ↪to_numpy(),d1['Crop_2_yield_sole'].to_numpy()])
print("Only %s/%s entries have valid S_i in dataset 1"%(len(idxs), len(d1)))
idxs = project.get_valid_indices_all_vars([d2['Crop_1_yield_sole'].
 ↪to_numpy(),d2['Crop_2_yield_sole'].to_numpy()])
print("Only %s/%s entries have valid S_i in dataset 2"%(len(idxs), len(d2)))
```

```
Only 934/934 entries have valid S_i in dataset 1
Only 956/1544 entries have valid S_i in dataset 2
```

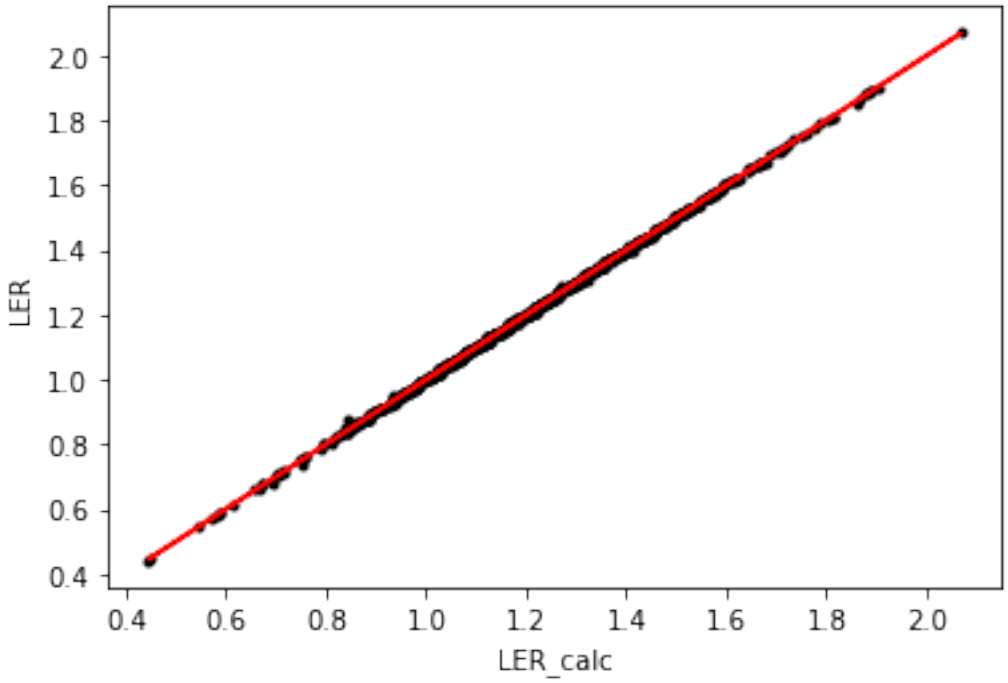
2. Calculer les LERs pour les deux datasets en utilisants les entrées valides. Tracer la distribution des LERs et comparer les LERs ainsi calculés avec ceux des papiers (LER_tot). Fitter un modèle linéaire en utilisant sckit-learn.

[4]:
```
LER_calc_1 = project.compute_LER(d1)
LER_calc_2 = project.compute_LER(d2)
```

```
/home/kodda/Dropbox/p2pflab/kaku/L3INFO/TME_agri/all_sessions/session1.py:28:
RuntimeWarning: invalid value encountered in divide
  return IY1/SY1+IY2/SY2
```

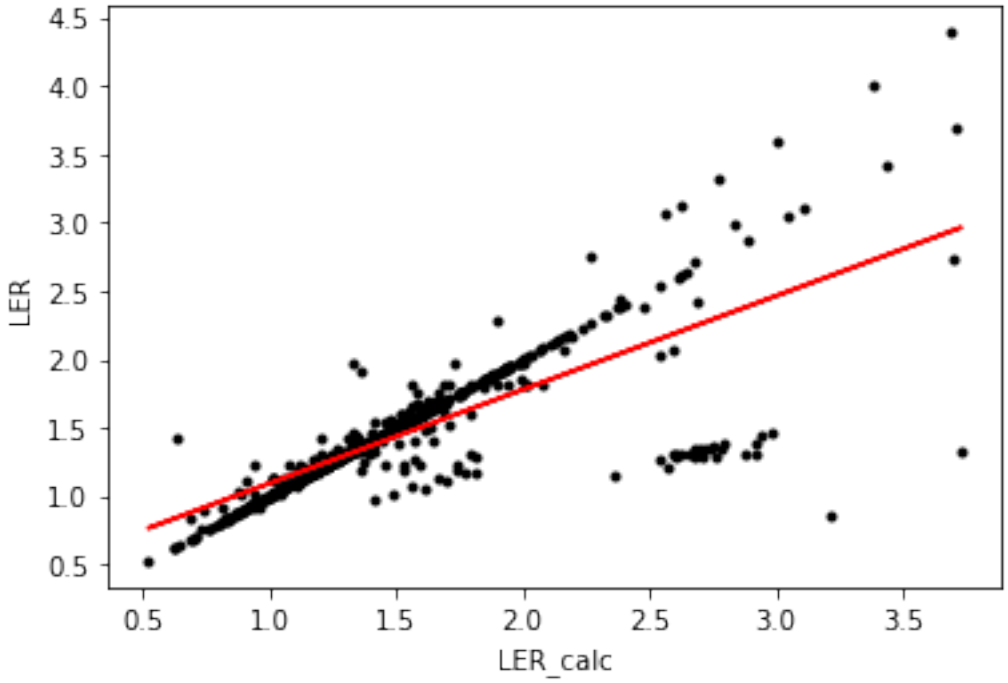
[5]:
```
project.plot_LERs(LER_calc_1, d1['LER_tot'].to_numpy())
```

```
RMSE :  1.6329509061755673e-05
R2 :  0.9996849942993267
```



[6]:
```
project.plot_LERs(LER_calc_2, d2['LER_tot'].to_numpy())
```

```
RMSE :  0.07852940138970738
R2 :  0.6108283423933467
```



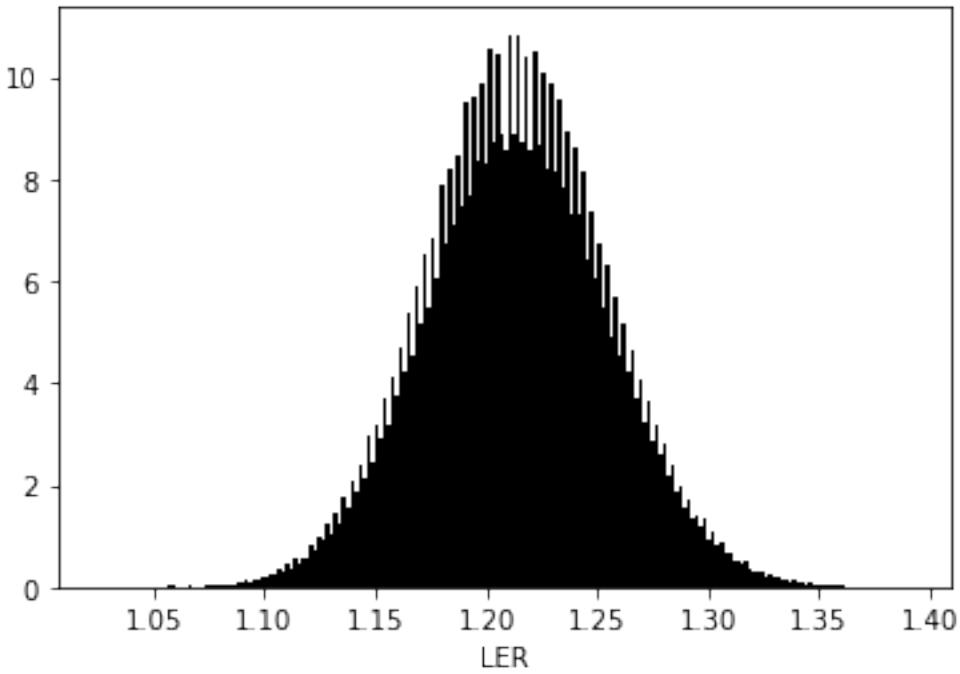
3. Calculer la moyenne et la variance des LERs. Montrer que l'estimateur naïf pour la variance est biaisé. Calculer l'intervalle de confiance à 95%. Tester avec statsmodels l'hypothèse LER>1.

[7]:
```
mLER, sLER, inter = project.compute_mean_std_inter(d1['LER_tot'].to_numpy())
print("Dataset 1 Mean: %s, Std: %s, interval .95: [%s, %s]"%(mLER, sLER,␣
 ↪inter[0], inter[1]))

mLER, sLER, inter = project.compute_mean_std_inter(d2['LER_tot'].to_numpy())
print("Dataset 2 Mean: %s, Std: %s, interval .95: [%s, %s]"%(mLER, sLER,␣
 ↪inter[0], inter[1]))

H0 = project.testH0(d1['LER_tot'].to_numpy())
```

```
Dataset 1 Mean: 1.21389721627409, Std: 0.22768129734656678, interval .95:
[1.1992952954030875, 1.2284991371450924]
Dataset 2 Mean: 1.3941361746361747, Std: 0.4609378570563445, interval .95:
[1.3650081580147109, 1.4232641912576385]
```

4. Etudier la distribution des moyennes pour différents échantillons.

[8]:
```
project.plot_dist_mean(d1['LER_tot'].to_numpy())
```



4. Classer les cultures suivant la probabilité qu'une expérience montre un rendement accru pour

l'association de culture (on ne considérera que les cultures qui interviennent dans plus de 10 expérience).

[9]:
```
project.get_sorted_crops(d2)
```

[9]:
```
[('Cabbage', 0.9393939393939388),
 ('Cauliflower', 0.9354838709677413),
 ('Strawberry', 0.8571428571428569),
 ('Safed musli', 0.833333333333333),
 ('Carrot', 0.703703703703703),
 ('Pepper', 0.6891891891891884),
 ('Mustard', 0.6666666666666666),
 ('Okra', 0.6216216216216215),
 ('Tomato', 0.6121212121212124),
 ('Fenugreek', 0.6086956521739129),
 ('Radish', 0.5833333333333333),
 ('Lablab bean', 0.5769230769230768),
 ('Apple', 0.513513513513513),
 ('Basil', 0.5000000000000001),
 ('Squash', 0.49999999999999994),
 ('Cucumber', 0.4893617021276593),
 ('Eggplant', 0.4500000000000001),
 ('Fava bean', 0.43661971830985896),
 ('Lettuce', 0.4117647058823534),
 ('Banana', 0.4000000000000001),
 ('Collard', 0.3846153846153846),
 ('Onion', 0.3684210526315787),
 ('Beetroot', 0.3658536585365854),
 ('Maize', 0.3397435897435888),
 ('Groundnut', 0.32258064516129026),
 ('Potato', 0.32075471698113206),
 ('Millet', 0.2962962962962963),
 ('Pea', 0.29508196721311475),
 ('Peanut', 0.2708333333333333),
 ('Bean', 0.2696245733788393),
 ('Cowpea', 0.2453703703703701),
 ('Soybean', 0.23076923076923078),
 ('Pigeon pea', 0.2236842105263157),
 ('Cassava', 0.19047619047619047),
 ('Pumpkin', 0.16666666666666666),
 ('Wheat', 0.14814814814814814),
 ('Garlic', 0.09090909090909091),
 ('Mung bean', 0.08333333333333333),
 ('Pearl millet', 0.044444444444444446),
 ('Cluster bean', 0.030303030303030304),
 ('Black gram', 0),
 ('White leadtree', 0),
```

```
('Rice', 0)]
```

5. Déterminer les groupes de cultures qui s'associent bien (LER>1.8). On créera pour cela un graphe dont on isolera les composantes connexes en utilisant networkx.

[10]:
```
project.list_clusters(d2, th=1.8)
```

```
Cluster 0
-----------
Coffee arabica
Rubber
Banana
-----------
Cluster 1
-----------
Bean
Parsley
Carrot
Pepper
-----------
Cluster 2
-----------
Cassava
Turmeric
Xanthosoma
Sapota
-----------
Cluster 3
-----------
Durum wheat
Olive
-----------
Cluster 4
-----------
Jackfruit
Eggplant
-----------
Cluster 5
-----------
Mustard
Spinach
Strawberry
Fava bean
Onion
Fenugreek
Tomato
Pigeon pea
Radish
```

```
Lablab bean
Lettuce
Maize
Safed musli
Marigold
-----------
Cluster 6
-----------
Perennial ryegrass
Indian jujube
Rattan grass
-----------
```

[ ]: