Guide d'utilisateur du package ASDSVC

Simon Bélanger

2017-10-03

Introduction

Ce document présente les principales fonctionalités du package ASDSVC. Cette librairie a été développée pour calculer les réflectances pour des surfaces terrestres ou pour des surfaces aquatiques.

Nous prennons pour acquis que les données ASD auront été converties en format texte (ASCII) au préalable dans le logiciel **ViewSpec Pro** qui n'est pas décrit dans ce document.

Principe de la mesure

Surface terrestre

En télédétection multispectrale, on s'intéresse à la signature spectrale des surfaces qu'on exprime en terme de réflectance sepctrale (parfois appelée albédo). La réflectance est le rapport des éclairements réfléchi et incident:

$$R = \frac{E_{refl}}{E_{inc}} \tag{1}$$

Or en télédétection on mesure la luminance, soit une flux d'énergie (Joule/s ou watt, W) par unité d'aire (m^2) et par unité d'angle solide (sr^{-1}). Si on considère une surface lambertienne, on peut suposer que la luminance est égale dans toutes les directions. Ainsi l'intégrale de l'hémisphère étant égale à π sr^{-1}, on peut définir la réflectance en terme de :

$$\rho = \frac{\pi L}{E_{inc}} \tag{2}$$

L'ASD mesure la luminance, L, à l'aide d'un capteur dont le champs de visé est de 25°. Pour mesurer l'éclairement incident avec un instrument comme l'ASD, on utilise un réflecteur lambertien dont la réflectance est connu ($R_{spectralon}$), soit un paneau de spectralon. Le panneau de spectralon utilisé à l'UQAR possède un $R_{spectralon}$ de 98,5%. Ainsi on a

$$E_{inc} = \frac{\pi L_{spectralon}}{R_{spectralon}} = \frac{\pi L_{spectralon}}{0,985} \tag{3}$$

L'ASD mesure la luminance, L, pendant un certain temps qu'on appelle le temps d'intégration, T_{int}. Le T_{int}, c-à-d. la durée de la mesure en milliseconde (ms), varie selon la quantité de lumière qui entre dans l'ASD. Comme la luminance de l'ASD n'est pas étalonnée et les valeurs sont exprimées en "compte numérique" variant entre 0 et 65 535 (16 bits), il faut tenir compte du temps d'intégration dans les calculs de réflectance. En effet chaque mesure de L doit être divisée (ou normalisée) par T_{int}. La luminance normalisée est donc

$$\hat{L} = \frac{L}{T_{int}} \tag{4}$$

Si on combine les équations 2, 3 et 4, on obtient l'équation générale suivante:

$$\rho = \frac{(L_{surface}/T_{int}^{surface})R_{spectralon}}{(L_{spectralon}/T_{int}^{spectralon})} = \frac{\hat{L}_{surface}R_{spectralon}}{\hat{L}_{spectralon}} \tag{5}$$

Surface aquatique

Le calcul de la réflectance pour une surface aquatique diffère légèrement de la méthode décrite ci-dessus. En effet, la réflexion d'une surface aquatique est la somme de la réflexion spéculaire causée par l'interface air-mer et de la réflexion diffuse qui se produit dans la colonne d'eau. C'est cette dernière qu'on voudra mesurer. Mobley (1999) a décrit avec précision le problème physique de cette mesure. Le lecteur devrait lire cette publication afin de bien comprendre la complexité du problème.

Essentiellement, la réflectance d'une surface aquatique est

$$\rho_{surface} = \rho_w + \rho_{sky}$$

où ρ_w est la réflectance diffuse de l'eau et ρ_{sky} la réflectance du ciel dans la direction de visée du capteur. La quantité qu'on désire est ρ_w qui en terme de luminances mesurées par l'ASD s'écrit comme étant:

$$\rho_w = \pi \frac{L_w}{E_d} = \pi \frac{(\hat{L}_{surface} - \rho_{fresnel}(\theta_s, \theta_v, \Delta\phi, W_s)\hat{L}_{sky})}{(\hat{L}_{spectralon}/R_{spectralon})} \tag{6}$$

où L_w est la lumiance émergeant de l'eau, $\rho_{fresnel}$ est le coefficient de réflectance de l'interface air-mer et $\hat{L}_{sky}$ est la luminance du ciel dans la direction d'où provient la lumière qui sera réfléchie par l'interface air-mer vers le capteur. Ainsi pour déterminer ρ_w il faut faire trois mesures avec l'ASD: $\hat{L}_{surface}, \hat{L}_{sky}, \hat{L}_{spectralon}$. Par ailleurs, le $\rho_{fresnel}$ est connu, par la loi de Fresnel, pour une surface d'eau parfaitement plane. Dans ces conditions $\rho_{fresnel}$ varie simplement avec l'angle zénithal de visé du capteur θ_v. Dans les conditions d'observations typiques en bateau, cependant, le coefficient $\rho_{fresnel}$ varie en fonction non seulement de θ_v, mais également de l'angle zénithal solaire θ_s, de la différence d'azimuth entre le soleil et la direction de visée, $\Delta\phi$ et, finalement, de la rugosité de la surface (des vagues) qui varie en fonction de la vitesse du vent, W_s. Mobley (1999), et plus récemment Mobley (2015), a utilisé des modèles numériques de monte carlo pour calculer les valeurs de $\rho_{fresnel}$ pour toutes les combinaisons de $\theta_s, \theta_v, \Delta\phi, W_s$. Ces valeurs sont stockées dans une table de correspondance (dit en anglais *look-up-table* ou LUT). La figure suivante montre comment $\rho_{fresnel}$ varie en fonction de la géométrie de visée et la vitesse du vent pour différente hauteur du soleil dans le ciel.

Malgré toutes les précautions prises sur le terrain, la mesure de réflectance au-dessus de l'eau comporte une erreur résiduelle qui peut, dans certain cas, être relativement importante. Cette erreur résiduelle est due à l'erreur sur $\rho_{fresnel}$ causé par l'environnement et d'autres artéfacts comme la présence d'écume, la réflexion spéculaire du soleil, la précision de la géométrie de visé (souvent approximative avec l'ASD), le champs de visé instantané du capteur (e.g. 25°), etc. Ainsi on doit considérer que

$$\rho_w(\lambda) = \rho_w^M(\lambda) - \epsilon \tag{7}$$

où $\rho_w^M(\lambda)$ sont les réflectances marines calculées avec l'équation 6 et la valeur de $\rho_{fresnel}$ provenant de la LUT de Mobley (2015), et ϵ est une erreur résiduelle.

La première méthode pour estimer ϵ considère l'hypothèse du pixel noir qui est strictement valide dans les eaux claires optiquement profondes. On pourra ainsi supposer que la réflectance est égale à 0 dans le proche infrarouge (PIR): $\epsilon = \rho_w^M(\lambda_{PIR})$. On peut prendre une λ_{PIR} autour de 870 à 900 nm où l'absorption de l'eau pure est supérieure à $4.7m^{-1}$.

Les deux autres méthodes sont basées sur le spectre de similarité dans le proche infrarouge (PIR). En effet on peut considérer que la forme spectrale de la réflectance marine dans le PIR est invariable dans les eaux relativement peu à moyennement turbides (voir Ruddick et al. (2006)). Ainsi on peut considérer que les rapports de réflectance entre deux longueurs d'onde du PIR (λ_1, λ_2) sont connus. Ruddick, Cauwer, and Mol (2005) ont démontré que l'erreur résiduelle, ϵ, peut être calculée algébriquement de cette matière:

$$\epsilon = \frac{\alpha_{1,2}\rho_w^M(\lambda_2) - \rho_w^M(\lambda_1)}{\alpha_{1,2} - 1} \tag{8}$$

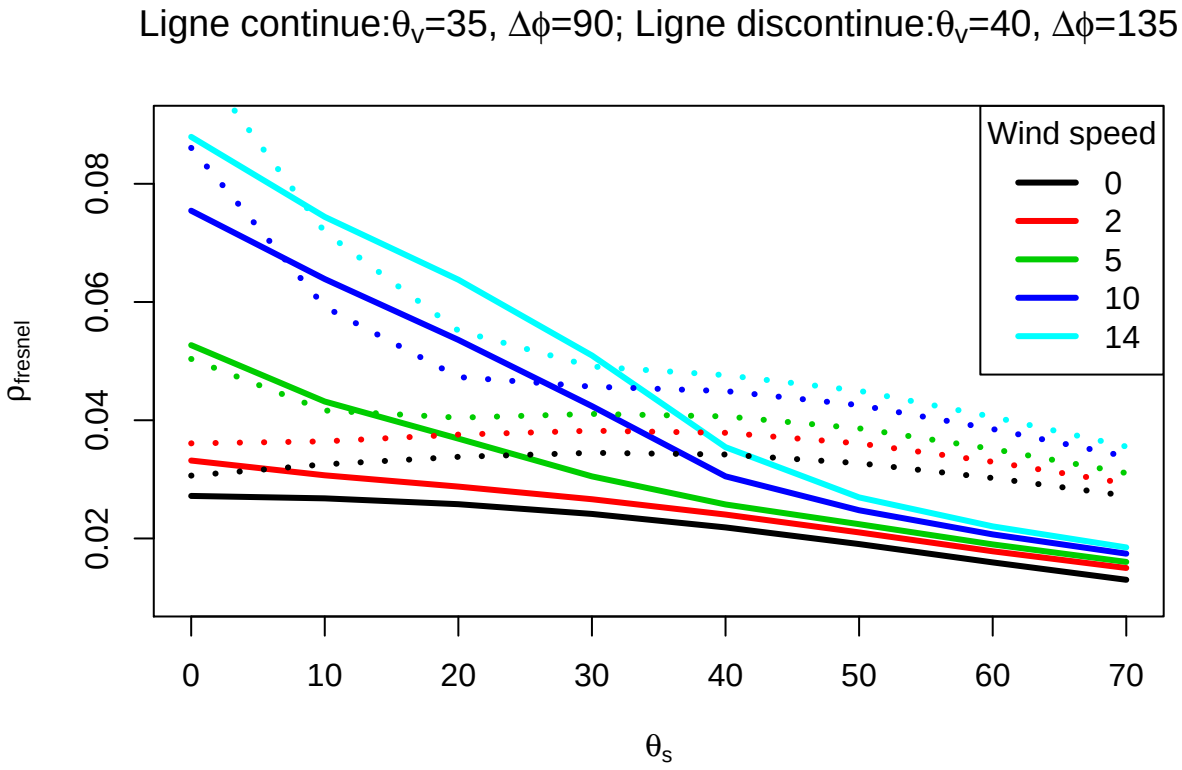


Figure 1: Variabilité du coefficient de réflexion de l'interface air-mer

où $\alpha_{1,2}$ est le ratio de réflectance marine connu entre deux longueurs d'ondes. Ruddick et al. (2005) suggèrent de prendre soit le couple λ_1=720 et λ_2=780 ($\alpha_{1,2} = 2.35$) ou le couple λ_1=780 et λ_2=870 nm ($\alpha_{1,2} = 1.91$) pour corriger les spectres de réflectance.

Une dernière approche est aussi considérée pour calculer ρ_w. Elle estime le $\rho_{fresnel}$ sans passer par la LUT de Mobley (2015) en prennant l'hypothèse du pixel noir dans le PIR (black pixel, BP, i.e. que $\rho_w(\lambda_{PIR}) == 0$).

$$\begin{aligned} \rho_{sky}(\lambda_{PIR}) &= \rho_{surface}(\lambda_{PIR}) \\ \rho^{PIR}_{fresnel}\pi\frac{\hat{L}_{sky}(\lambda_{PIR})}{E_d} &= \pi\frac{\hat{L}_{surface}(\lambda_{PIR})}{E_d} \\ \rho^{PIR}_{fresnel} &= \frac{\hat{L}_{surface}(\lambda_{PIR})}{\hat{L}_{sky}(\lambda_{PIR})} \end{aligned} \tag{9}$$

De manière similaire, l'absorption dans l'UV est parfois très élevée dans les eaux côtières en présence de CDOM. La réflectance est souvent négligeable à ce domaine spectral. L'ASD permet de mesurer jusqu'à 325 nn, soit au début du domaine UV-A.

$$\begin{aligned} \rho_{sky}(\lambda_{UV}) &= \rho_{surface}(\lambda_{UV}) \\ \rho^{UV}_{fresnel}\pi\frac{\hat{L}_{sky}(\lambda_{UV})}{E_d} &= \pi\frac{\hat{L}_{surface}(\lambda_{UV})}{E_d} \\ \rho^{UV}_{fresnel} &= \frac{\hat{L}_{surface}(\lambda_{UV})}{\hat{L}_{sky}(\lambda_{UV})} \end{aligned} \tag{10}$$

Avec cette méthode, les $\rho^{PIR}_{fresnel}$ et $\rho^{UV}_{fresnel}$ sont estimés indépendamment de la géométrie de visée, du vent et de la hauteur du soleil. Elles peuvent s'avérer meilleures que la méthode théotique quand les conditions ne sont pas idéales comme un ciel variable avec du vent où la théorie colle difficilement avec la réalité.

Ces méthodes de correction seront invalides dès qu'on se retrouvera en présence de végétation flottant sur l'eau ou dans les eaux peu profondes ou ni l'hypothèse du pixel noir, ni l'hypothèse du spectre de similarité ne sont valides.

Acquisition des données ASD sur le terrain

Les données ASD sont acquises sur le terrain avec le logiciel RS^3. Afin de réduire le ratio signal-sur-bruit (*signal-to-noise ratio* ou *SNR*) on ajuste le temps d'intégration pour chaque surface et on collecte plusieurs spectres pour la même surface. Ceux-ci seront moyennés par la suite et les spectres douteux éliminés. Dans RS^3, on doit spécifier la **base du nom des fichiers** (*basename*) qui seront générés par le logiciel. Ce dernier ajoute automatiquement un nombre à la suite du *basename* de manière incrémentale. Supposons qu'on veut mesurer la réflectance de l'eau de la station X, on prendra sur le terrain 10 spectres du spectralon, 10 spectres du ciel et 10 spectres de la surface. Avant de procéder, on changera le *basename* pour "StationX_" dans le menu *save spectrum* du logiciel RS^3 on on indiquera au logiciel de faire 10 mesures à chaque fois qu'on démare l'acquisition. Donc pour les trois mesures le luminance, le logiciel générera ainsi au total 30 fichiers qui seront nommés ainsi:

- pour le spectralon, les fichiers sont nommés **StationX_00000.asd** jusqu'à **StationX_00009.asd**;
- pour le ciel, les fichiers sont nommés **StationX_00010.asd** jusqu'à **StationX_00019.asd**;
- pour la surface, les fichiers sont nommés **StationX_00020.asd** jusqu'à **StationX_00029.asd**.

Il est absolument essentiel de noter ces informations sur le terrain et les transcrire dans un fichier log. Pour les surfaces aquatiques, nouss avons pris l'habitude d'inclure dans le fichier log les colonnes suivantes:

- **lat**: est la latitude en degré;
- **lon**: est la longitude en degré;
- **basename**: est la base du nom des fichier de l'ASD (ex: "StationX_");
- **ID**: est le nom de la station du transect (il peut donc être différent du *basename*);
- **Lpanel_start**: est le numéro du premier fichier de la série de mesure de luminance du panneau de spectralon (ex: 0 dans l'exemple ci-haut);

- **Lpanel_end**: est le numéro du dernier fichier de la série de mesure de luminance du panneau de spectralon (ex: 9dans l'exemple ci-haut);
- **Lsky_start**: est le numéro du premier fichier de la série de mesure de luminance du ciel (ex: 10 dans l'exemple ci-haut);

- **Lsky_end**: est le numéro du dernier fichier de la série de mesure de luminance du ciel (ex: 19 dans l'exemple ci-haut);

- **Ltot_start**: est le numéro du premier fichier de la série de mesure de luminance de la surface totale (ex: 20 dans l'exemple ci-haut);
- **Ltot_end**: est le numéro du dernier fichier de la série de mesure de luminance de la surface totale (ex: 29 dans l'exemple ci-haut);

- **ThetaV**: est l'angle zénithal de visé, θ_v, généralement entre 35 et 40 degrés;
- **Dphi**: est la différence d'azimuth entre l'azimuth du soleil et l'azimuth de l'ASD, $\Delta\phi$, généralement entre 90 et 135 degrés;
- **Windspeed** est la vitesse du vent;
- **Wind.units** est l'unité de la vitesse du vent;

Le fichier log comprend deux autres champs qui seront utilisés durant le traitement des données. On doit les inclure quand même quand un veut utiliser les fontionnalités du package pour le traitement semi-automatisé du package (voir section **Traitement en batch** ci-bas)

- **quantile.prob** est la valeur du quantile au-delà duquel les spectres seront éliminés (varie entre 0.25 et 1). Il a pour but d'éliminer les spectres qui seraient contaminés par la réflexion speculaire du soleil. Par exemple, une valeur de 0.75 éliminera les spectres au-delà du troisième quantile. Une valeur de 0.5

éliminera les spectres au-delà du deuxième quantile, soit les spectres qui sont au dessus de la moyenne de tous les spectres (voir plus bas pour plus de détails).

- **rhow.Method** est la méthode de correction de la réflexion spéculaire qui sera retenue **à la fin du traitement** des données. Cette information sera utilisée pour générer la base de données validées. Sept options sont possibles: + 0 : Méthode de Mobley pour $\rho_{Fresnel}$ sans correction dans le PIR; + 1 : Méthode de Mobley pour $\rho_{Fresnel}$ avec la coorection du pixel noir; + 2 : Méthode de Mobley pour $\rho_{Fresnel}$ avec la coorection utilisant le spectre de similarité de de Ruddick et al. (2005) avec λ_1=720 et λ_2=780 (eq. 8); + 3 : Méthode de Mobley pour $\rho_{Fresnel}$ avec la coorection utilisant le spectre de similarité de de Ruddick et al. (2005) avec λ_1=780 et λ_2=870 (eq. 8); + 4 : Méthode du pixel noir dans le PIR pour le $\rho_{Fresnel}$ (eq. 9); + 5 : Méthode du pixel noir dans le UV pour le $\rho_{Fresnel}$ (eq. 10); + 999 : quand on juge que les données ne passent pas le contrôle de qualité.

Idéalement, il faut également inclure dans le log la position en latitude et longitude puisque cette information est aussi essentiel pour calculer la hauteur du soleil, θ_s, au moment de la mesure. Toutes ces observations auxiliaires seront éventuellement nécessaire dans le traitement des données.

Installation de la librairie `asd`

La librarie `asd` se trouve que la plateforme GitHub en ligne et peut s'installer avec la commande `install_github` de la librarie `devtools` qui doit également être installée.

```
devtools::install_github("belasi01/asd")
```

Principales fonctions de la librairie `asd`

La librairie `asd` comprend des fonctionallités qui permettent à l'utilisateur de manipuler les données relativement facilement.

Lecture des données ASD

####La fonction `read.ASD()` Elle permet de lire les fichiers ASCII de l'ASD qui ont été produits par le logiciel *ViewSpec pro*. Elle retourne un objet de type 'list' qui contient:

- **L.norm**: est un vecteur de luminance normalisée par le temps d'intégration;
- **waves**: est un vecteur de longueurs d'onde;
- **DateTime**: est la date et l'heure de l'acquisition en format POSIXct qui correspond à la date et l'heure de l'ordinateur au moment de la mesure, généralement en UTC;
- **IntTime**: est le temps d'intégration en millisecondes.

Voici un exemple:

```
nomASD= "~/OneDrive - UQAR/data/CHOne-2-BSI/L1/2017/L1/ASD/20170623_UQAR/bsi_7_4_00000.asd.txt"
asd = read.ASD(nomASD)
```

```
## [1] "~/OneDrive - UQAR/data/CHOne-2-BSI/L1/2017/L1/ASD/20170623_UQAR/bsi_7_4_00000.asd.txt"
```

```
str(asd)
```

```
## List of 4
##  $ L.norm  : num [1:751] 19.1 24 30.1 37.5 45.8 ...
##  $ waves   : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##  $ DateTime: POSIXct[1:1], format: "2017-06-23 15:15:42"
##  $ IntTime : num 34
```

Les fonctions `average.ASD.replicats()` et `generate.file.names.ASD()`

Comme expliqué ci-dessus, on fait souvent plusieurs mesures d'une même surface. La fonction `average.ASD.replicats()` permettra de lire plusieurs fichiers, en appelant la fonction `read.ASD()`, et

stocker l'information dans un objet unique qui contiendra tous les spectres de luminance de la surface. La fonction `average.ASD.replicats()` prend un seul argument qui est un vecteur qui contient les noms des fichiers qu'on veut moyenner. Ce vecteur de noms de fichier sera éventuellement généré avec les informations trouvées dans le fichier log décrit ci-haut avec la fonction `generate.file.names.ASD()`. Voyons un exemple concret:

```
Lpanel_start = 0
Lpanel_end = 19
basename = "bsi_7_4"
setwd("~/OneDrive - UQAR/data/CHOne-2-BSI/L1/2017/L1/ASD/20170623_UQAR/")
nomsfichiersASD = generate.file.names.ASD(basename, Lpanel_start,Lpanel_end)
print(nomsfichiersASD)
```

```
##  [1] "bsi_7_4_00000.asd.txt" "bsi_7_4_00001.asd.txt" "bsi_7_4_00002.asd.txt"
##  [4] "bsi_7_4_00003.asd.txt" "bsi_7_4_00004.asd.txt" "bsi_7_4_00005.asd.txt"
##  [7] "bsi_7_4_00006.asd.txt" "bsi_7_4_00007.asd.txt" "bsi_7_4_00008.asd.txt"
## [10] "bsi_7_4_00009.asd.txt" "bsi_7_4_00010.asd.txt" "bsi_7_4_00011.asd.txt"
## [13] "bsi_7_4_00012.asd.txt" "bsi_7_4_00013.asd.txt" "bsi_7_4_00014.asd.txt"
## [16] "bsi_7_4_00015.asd.txt" "bsi_7_4_00016.asd.txt" "bsi_7_4_00017.asd.txt"
## [19] "bsi_7_4_00018.asd.txt" "bsi_7_4_00019.asd.txt"
```

On peut voir que la fonction `generate.file.names.ASD()` a pris trois arguments et a généré le vecteur avec les noms de fichier qu'on veut mettre ensemble. Dans cet exemple il s'agit des noms des fichiers pour une mesure de spectralon.

La ligne de code suivante lance la fonction `average.ASD.replicats()` qui retournera un objet de type liste, Lp, qui compte six variables, dont la matrice (Lum) avec tous les spectres de luminance normalisée.

```
Lp = average.ASD.replicats(nomsfichiersASD)
```

```
## [1] "bsi_7_4_00000.asd.txt"
## [1] "bsi_7_4_00000.asd.txt"
## [1] "bsi_7_4_00001.asd.txt"
## [1] "bsi_7_4_00002.asd.txt"
## [1] "bsi_7_4_00003.asd.txt"
## [1] "bsi_7_4_00004.asd.txt"
## [1] "bsi_7_4_00005.asd.txt"
## [1] "bsi_7_4_00006.asd.txt"
## [1] "bsi_7_4_00007.asd.txt"
## [1] "bsi_7_4_00008.asd.txt"
## [1] "bsi_7_4_00009.asd.txt"
## [1] "bsi_7_4_00010.asd.txt"
## [1] "bsi_7_4_00011.asd.txt"
## [1] "bsi_7_4_00012.asd.txt"
## [1] "bsi_7_4_00013.asd.txt"
## [1] "bsi_7_4_00014.asd.txt"
## [1] "bsi_7_4_00015.asd.txt"
## [1] "bsi_7_4_00016.asd.txt"
## [1] "bsi_7_4_00017.asd.txt"
## [1] "bsi_7_4_00018.asd.txt"
## [1] "bsi_7_4_00019.asd.txt"
```

```
str(Lp)
```

```
## List of 6
##  $ Lum      : num [1:751, 1:20] 19.1 24 30.1 37.5 45.8 ...
##  $ L.norm   : num [1:751] 20.2 25.6 31.7 39.4 48.3 ...
```

```
##  $ L.norm.sd: num [1:751] 0.691 0.808 0.763 0.888 1.099 ...
##  $ waves    : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##  $ DateTime : POSIXct[1:20], format: "2017-06-23 15:15:42" "2017-06-23 15:15:42" ...
##  $ IntTime  : num [1:20] 34 34 34 34 34 34 34 34 34 34 ...
```

Calcul de la réflectance spectrale pour les surfaces aquatiques

Dans cette section nous verrons deux fonctions de la librairie qui serviront à calculer la réflectance de l'eau, ρ_w. La première sert à assembler toutes les données nécessaires au calcul de ρ_w et la seconde applique les équations décrites ci-haut (6 à 10).

La fonction `merge.ASD.radiances.for.rhow()`

Cette fonction sert à assembler toutes les données nécessaires au calcul de ρ_w, soit les luminances du panneau de spectralon, du ciel et de la surface, ainsi que les données auxiliaires comme la géométrie de visé, la vitesse du vent, la position GPS et l'heure du GPS. La fonction calculera la position du soleil avec l'information fourni. Elle génère un objet de type *list* qui contiendra toutes les données requises pour le calcul de ρ_w.

Reprenons l'exemple précédant:

```
# Lecture des données ASD du panneau de spectralon, du ciel et de la surface
# Notez ici que l'arguement de average.ASD.replicats() est la fonction
# qui produit la liste des noms de fichiers, generate.file.names.ASD()
Lpanel=average.ASD.replicats(generate.file.names.ASD("bsi_7_4", 0,19))
```

```
## [1] "bsi_7_4_00000.asd.txt"
## [1] "bsi_7_4_00000.asd.txt"
## [1] "bsi_7_4_00001.asd.txt"
## [1] "bsi_7_4_00002.asd.txt"
## [1] "bsi_7_4_00003.asd.txt"
## [1] "bsi_7_4_00004.asd.txt"
## [1] "bsi_7_4_00005.asd.txt"
## [1] "bsi_7_4_00006.asd.txt"
## [1] "bsi_7_4_00007.asd.txt"
## [1] "bsi_7_4_00008.asd.txt"
## [1] "bsi_7_4_00009.asd.txt"
## [1] "bsi_7_4_00010.asd.txt"
## [1] "bsi_7_4_00011.asd.txt"
## [1] "bsi_7_4_00012.asd.txt"
## [1] "bsi_7_4_00013.asd.txt"
## [1] "bsi_7_4_00014.asd.txt"
## [1] "bsi_7_4_00015.asd.txt"
## [1] "bsi_7_4_00016.asd.txt"
## [1] "bsi_7_4_00017.asd.txt"
## [1] "bsi_7_4_00018.asd.txt"
## [1] "bsi_7_4_00019.asd.txt"
```

```
Lsky=average.ASD.replicats(generate.file.names.ASD("bsi_7_4", 20,39))
```

```
## [1] "bsi_7_4_00020.asd.txt"
## [1] "bsi_7_4_00020.asd.txt"
## [1] "bsi_7_4_00021.asd.txt"
## [1] "bsi_7_4_00022.asd.txt"
## [1] "bsi_7_4_00023.asd.txt"
## [1] "bsi_7_4_00024.asd.txt"
## [1] "bsi_7_4_00025.asd.txt"
## [1] "bsi_7_4_00026.asd.txt"
```

```
## [1] "bsi_7_4_00027.asd.txt"
## [1] "bsi_7_4_00028.asd.txt"
## [1] "bsi_7_4_00029.asd.txt"
## [1] "bsi_7_4_00030.asd.txt"
## [1] "bsi_7_4_00031.asd.txt"
## [1] "bsi_7_4_00032.asd.txt"
## [1] "bsi_7_4_00033.asd.txt"
## [1] "bsi_7_4_00034.asd.txt"
## [1] "bsi_7_4_00035.asd.txt"
## [1] "bsi_7_4_00036.asd.txt"
## [1] "bsi_7_4_00037.asd.txt"
## [1] "bsi_7_4_00038.asd.txt"
## [1] "bsi_7_4_00039.asd.txt"
```

```
Ltot=average.ASD.replicats(generate.file.names.ASD("bsi_7_4", 40,99))
```

```
## [1] "bsi_7_4_00040.asd.txt"
## [1] "bsi_7_4_00040.asd.txt"
## [1] "bsi_7_4_00041.asd.txt"
## [1] "bsi_7_4_00042.asd.txt"
## [1] "bsi_7_4_00043.asd.txt"
## [1] "bsi_7_4_00044.asd.txt"
## [1] "bsi_7_4_00045.asd.txt"
## [1] "bsi_7_4_00046.asd.txt"
## [1] "bsi_7_4_00047.asd.txt"
## [1] "bsi_7_4_00048.asd.txt"
## [1] "bsi_7_4_00049.asd.txt"
## [1] "bsi_7_4_00050.asd.txt"
## [1] "bsi_7_4_00051.asd.txt"
## [1] "bsi_7_4_00052.asd.txt"
## [1] "bsi_7_4_00053.asd.txt"
## [1] "bsi_7_4_00054.asd.txt"
## [1] "bsi_7_4_00055.asd.txt"
## [1] "bsi_7_4_00056.asd.txt"
## [1] "bsi_7_4_00057.asd.txt"
## [1] "bsi_7_4_00058.asd.txt"
## [1] "bsi_7_4_00059.asd.txt"
## [1] "bsi_7_4_00060.asd.txt"
## [1] "bsi_7_4_00061.asd.txt"
## [1] "bsi_7_4_00062.asd.txt"
## [1] "bsi_7_4_00063.asd.txt"
## [1] "bsi_7_4_00064.asd.txt"
## [1] "bsi_7_4_00065.asd.txt"
## [1] "bsi_7_4_00066.asd.txt"
## [1] "bsi_7_4_00067.asd.txt"
## [1] "bsi_7_4_00068.asd.txt"
## [1] "bsi_7_4_00069.asd.txt"
## [1] "bsi_7_4_00070.asd.txt"
## [1] "bsi_7_4_00071.asd.txt"
## [1] "bsi_7_4_00072.asd.txt"
## [1] "bsi_7_4_00073.asd.txt"
## [1] "bsi_7_4_00074.asd.txt"
## [1] "bsi_7_4_00075.asd.txt"
## [1] "bsi_7_4_00076.asd.txt"
## [1] "bsi_7_4_00077.asd.txt"
```

```
## [1] "bsi_7_4_00078.asd.txt"
## [1] "bsi_7_4_00079.asd.txt"
## [1] "bsi_7_4_00080.asd.txt"
## [1] "bsi_7_4_00081.asd.txt"
## [1] "bsi_7_4_00082.asd.txt"
## [1] "bsi_7_4_00083.asd.txt"
## [1] "bsi_7_4_00084.asd.txt"
## [1] "bsi_7_4_00085.asd.txt"
## [1] "bsi_7_4_00086.asd.txt"
## [1] "bsi_7_4_00087.asd.txt"
## [1] "bsi_7_4_00088.asd.txt"
## [1] "bsi_7_4_00089.asd.txt"
## [1] "bsi_7_4_00090.asd.txt"
## [1] "bsi_7_4_00091.asd.txt"
## [1] "bsi_7_4_00092.asd.txt"
## [1] "bsi_7_4_00093.asd.txt"
## [1] "bsi_7_4_00094.asd.txt"
## [1] "bsi_7_4_00095.asd.txt"
## [1] "bsi_7_4_00096.asd.txt"
## [1] "bsi_7_4_00097.asd.txt"
## [1] "bsi_7_4_00098.asd.txt"
## [1] "bsi_7_4_00099.asd.txt"
```

```
# Les données ASD et les informations auxiliaires sont passées à la fonction suivante
ASDtot = merge.ASD.radiances.for.rhow(Ltot,
                                      Lsky,
                                      Lpanel,
                                      StationID="BSI 7_4",
                                      lat=50.17,
                                      lon=-66.40,
                                      DateTime=mean.POSIXct(Ltot$DateTime),
                                      ThetaV=35,
                                      Dphi=135,
                                      Windspeed=12.0) # ici la vitesse de vent est en m/s
str(ASDtot)
```

```
## List of 4
##  $ Ltot  :List of 6
##   ..$ Lum      : num [1:751, 1:60] 0.468 0.6 0.747 0.912 1.094 ...
##   ..$ L.norm   : num [1:751] 0.527 0.681 0.85 1.041 1.251 ...
##   ..$ L.norm.sd: num [1:751] 0.036 0.0463 0.0579 0.0719 0.0875 ...
##   ..$ waves    : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##   ..$ DateTime : POSIXct[1:60], format: "2017-06-23 15:17:42" "2017-06-23 15:17:44" ...
##   ..$ IntTime  : num [1:60] 2176 2176 2176 2176 2176 ...
##  $ Lsky  :List of 6
##   ..$ Lum      : num [1:751, 1:20] 12.8 16.6 20.6 25.1 30.1 ...
##   ..$ L.norm   : num [1:751] 12.7 16.3 20.5 25.1 30 ...
##   ..$ L.norm.sd: num [1:751] 0.17 0.215 0.2 0.231 0.3 ...
##   ..$ waves    : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##   ..$ DateTime : POSIXct[1:20], format: "2017-06-23 15:16:43" "2017-06-23 15:16:43" ...
##   ..$ IntTime  : num [1:20] 136 136 136 136 136 136 136 136 136 136 ...
##  $ Lpanel:List of 6
##   ..$ Lum      : num [1:751, 1:20] 19.1 24 30.1 37.5 45.8 ...
##   ..$ L.norm   : num [1:751] 20.2 25.6 31.7 39.4 48.3 ...
##   ..$ L.norm.sd: num [1:751] 0.691 0.808 0.763 0.888 1.099 ...
```

```
##   ..$ waves    : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##   ..$ DateTime : POSIXct[1:20], format: "2017-06-23 15:15:42" "2017-06-23 15:15:42" ...
##   ..$ IntTime  : num [1:20] 34 34 34 34 34 34 34 34 34 34 ...
##  $ anc   :List of 9
##   ..$ StationID: chr "BSI 7_4"
##   ..$ lat      : num 50.2
##   ..$ lon      : num -66.4
##   ..$ DateTime : POSIXct[1:1], format: "2017-06-23 15:19:02"
##   ..$ ThetaV   : num 35
##   ..$ Dphi     : num 135
##   ..$ Windspeed: num 12
##   ..$ ThetaS   : num 29.9
##   ..$ PhiS     : num 147
```

On peut voir que l'objet *ASDtot* rassemble toutes l'information nécessaire pour le calcul de la réflectance de l'eau. Notez qu'ici la date et l'heure passées à la fonction `merge.ASD.radiances.for.rhow()` est la moyenne des mesures de luminance de la surface (Ltot) dans le format POSIXct de R. C'est avec cette information que la position du soleil dans le ciel sera calculée. L'objet *ASDtot* est une liste qui contient elle-même quatre listes (Ltot, Lsky, Lpanel et anc).

la fonction `compute.ASD.rhow()`

La fonction `compute.ASD.rhow()` accepte trois arguments:

- Le premier, **raw.asd**, est l'objet retourné par la fonction `merge.ASD.radiances.for.rhow()`.
- Le deuxième,**rho.panel**, est la réflectance du spectralon qui est normalement connu.
- Le troisième, **quantile.prob**, est la valeur du quantile au-delà duquel les spectres seront éliminés (varie entre 0.25 et 1). Il a pour but d'éliminer les spectres qui seraient contaminés par la réflexion speculaire du soleil. Par exemple, une valeur de 0.75 éliminera les spectres au-delà du troisième quantile. Une valeur de 0.5 éliminera les spectres au-delà du deuxième quantile, soit les spectres qui sont au-dessus de la moyenne de tous les spectres.

La fonction `compute.ASD.rhow()` calculera la réflectance de l'eau tel que décrit ci-haut. Elle fera :

- la moyenne des spectres de luminance du spectralon, du ciel et de la surface. Pour cette dernière on utilisera une moyenne inter-quantile qui sera contrôlé par un argument de la fonction;
- estimera le coefficient de réflexion spéculaire, $\rho_{fresnel}$. Si le ciel est découvert, $\rho_{fresnel}$ est interpolé à partir des informations fournies et de la table de Mobley (2015), si non (ciel couvert) une valeure constante de 0.0256 est utilisée (voir Ruddick et al L&O 2006), et enfin $\rho_{fresnel}$ est estimé à partir de l'hypotèse du pixel noir dans le PIR (eq 9; méthode 4) ou dans l'UV (eq 10; méthode 5), ou les deux (méthode 6; en interpolant les valeurs de $\rho_{fresnel}$ entre le PIR et l'UV). La méthode 7 est applicable quand les données du C-OPS sont disponibles en estimant $\rho_{fresnel}$ en prennant la mesure de réflectance de l'eau mesurée par le C-OPS à deux longueurs d'ondes (PIR et UV) et en interpolant entre les deux (comme pour la méthode 6). Une dernière méthode (8) a été proposé par Kutser et al. (2013) et se base sur les mêmes hypothèses que pour la méthode 6;
- corrigera le spectre de réflectance pour la contamination résiduelle selon les méthodes qui ont été décrites ci-dessus.

Exemple de la structure de la liste de données produites par la fonction `compute.ASD.rhow()`:

```
rhow = compute.ASD.rhow(ASDtot, 0.985, quantile.prob = 0.75)
```

```
## [1] "Averaging Radiance spcetra..."
## [1] "Compute Sky and Surface reflectance and apply a smoothing function"
## [1] "Cloudy sky, Use rho = 0.0256"
## [1] "Apply NIR corrections"
## [1] "Begining the Kutser correction for glint"
```

```
## [1] "Wavelength and data at UV and NIR binned"
## [1] "Starting the NLS"
## [1] "Kutser correction finished"
## List of 31
##  $ waves           : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##  $ rhow            : num [1:751] 0.00964 0.00986 0.00993 0.00986 0.0097 ...
##  $ rhow.NULL       : num [1:751] 0.0065 0.00672 0.00679 0.00671 0.00656 ...
##  $ rhow.SIMILARITY1: num [1:751] 0.00351 0.00373 0.0038 0.00373 0.00357 ...
##  $ rhow.SIMILARITY2: num [1:751] 0.00695 0.00717 0.00724 0.00717 0.00701 ...
##  $ rhow.NIR        : num [1:751] -3.05e-04 -3.92e-05 7.82e-05 5.87e-05 -4.07e-05 ...
##  $ rhow.UV         : num [1:751] -0.00158 -0.00131 -0.00118 -0.0012 -0.00129 ...
##  $ rhow.UV.NIR     : num [1:751] -0.00168 -0.0014 -0.00127 -0.00128 -0.00137 ...
##  $ rhow.COPS       : num NA
##  $ rhow.Kutser     : num [1:751] 0.0026 0.00282 0.00287 0.00278 0.0026 ...
##  $ rhow.Jiang      : num [1:751] 0.00492 0.00515 0.00523 0.00516 0.00501 ...
##  $ rho.sky         : num 0.0256
##  $ rho.sky.NIR     : num 0.0415
##  $ rho.sky.UV      : num 0.0435
##  $ rho.sky.UV.NIR  : num [1:751] 0.0437 0.0437 0.0437 0.0437 0.0437 ...
##  $ rho.sky.COPS    : logi NA
##  $ Lpanel          : num [1:751, 1:16] 18.8 24.2 30.9 38.4 46.6 ...
##  $ Lt              : num [1:751, 1:39] 0.533 0.69 0.852 1.039 1.249 ...
##  $ Li              : num [1:751, 1:16] 12.8 16.6 20.6 25.1 30.1 ...
##  $ Ed.mean         : num [1:751] 64.7 82.1 101.6 126 154.7 ...
##  $ Ed.sd           : num [1:751] 1.91 2.22 1.92 2.16 2.49 ...
##  $ Lt.mean         : num [1:751] 0.525 0.679 0.846 1.036 1.244 ...
##  $ Lt.sd           : num [1:751] 0.0314 0.0398 0.0509 0.063 0.0754 ...
##  $ Li.mean         : num [1:751] 12.7 16.3 20.5 25.1 30 ...
##  $ Li.sd           : num [1:751] 0.148 0.203 0.198 0.23 0.28 ...
##  $ ix.Lt.good      : int [1:39] 5 6 7 8 9 10 11 12 13 14 ...
##  $ ix.Lsky.good    : int [1:16] 1 4 5 6 7 8 9 10 11 12 ...
##  $ ix.Lpanel.good  : int [1:16] 3 4 5 6 7 8 9 10 11 12 ...
##  $ DateTime        : POSIXct[1:1], format: "2017-06-23 15:18:49"
##  $ anc             :List of 9
##   ..$ StationID: chr "BSI 7_4"
##   ..$ lat      : num 50.2
##   ..$ lon      : num -66.4
##   ..$ DateTime : POSIXct[1:1], format: "2017-06-23 15:19:02"
##   ..$ ThetaV   : num 35
##   ..$ Dphi     : num 135
##   ..$ Windspeed: num 12
##   ..$ ThetaS   : num 29.9
##   ..$ PhiS     : num 147
##  $ CLEARSKY        : logi FALSE
```

```
str(rhow)
```

```
## List of 31
##  $ waves           : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##  $ rhow            : num [1:751] 0.00964 0.00986 0.00993 0.00986 0.0097 ...
##  $ rhow.NULL       : num [1:751] 0.0065 0.00672 0.00679 0.00671 0.00656 ...
##  $ rhow.SIMILARITY1: num [1:751] 0.00351 0.00373 0.0038 0.00373 0.00357 ...
##  $ rhow.SIMILARITY2: num [1:751] 0.00695 0.00717 0.00724 0.00717 0.00701 ...
##  $ rhow.NIR        : num [1:751] -3.05e-04 -3.92e-05 7.82e-05 5.87e-05 -4.07e-05 ...
##  $ rhow.UV         : num [1:751] -0.00158 -0.00131 -0.00118 -0.0012 -0.00129 ...
```

```
##  $ rhow.UV.NIR     : num [1:751] -0.00168 -0.0014 -0.00127 -0.00128 -0.00137 ...
##  $ rhow.COPS       : num NA
##  $ rhow.Kutser     : num [1:751] 0.0026 0.00282 0.00287 0.00278 0.0026 ...
##  $ rhow.Jiang      : num [1:751] 0.00492 0.00515 0.00523 0.00516 0.00501 ...
##  $ rho.sky         : num 0.0256
##  $ rho.sky.NIR     : num 0.0415
##  $ rho.sky.UV      : num 0.0435
##  $ rho.sky.UV.NIR  : num [1:751] 0.0437 0.0437 0.0437 0.0437 0.0437 ...
##  $ rho.sky.COPS    : logi NA
##  $ Lpanel          : num [1:751, 1:16] 18.8 24.2 30.9 38.4 46.6 ...
##  $ Lt              : num [1:751, 1:39] 0.533 0.69 0.852 1.039 1.249 ...
##  $ Li              : num [1:751, 1:16] 12.8 16.6 20.6 25.1 30.1 ...
##  $ Ed.mean         : num [1:751] 64.7 82.1 101.6 126 154.7 ...
##  $ Ed.sd           : num [1:751] 1.91 2.22 1.92 2.16 2.49 ...
##  $ Lt.mean         : num [1:751] 0.525 0.679 0.846 1.036 1.244 ...
##  $ Lt.sd           : num [1:751] 0.0314 0.0398 0.0509 0.063 0.0754 ...
##  $ Li.mean         : num [1:751] 12.7 16.3 20.5 25.1 30 ...
##  $ Li.sd           : num [1:751] 0.148 0.203 0.198 0.23 0.28 ...
##  $ ix.Lt.good      : int [1:39] 5 6 7 8 9 10 11 12 13 14 ...
##  $ ix.Lsky.good    : int [1:16] 1 4 5 6 7 8 9 10 11 12 ...
##  $ ix.Lpanel.good  : int [1:16] 3 4 5 6 7 8 9 10 11 12 ...
##  $ DateTime        : POSIXct[1:1], format: "2017-06-23 15:18:49"
##  $ anc             :List of 9
##   ..$ StationID: chr "BSI 7_4"
##   ..$ lat      : num 50.2
##   ..$ lon      : num -66.4
##   ..$ DateTime : POSIXct[1:1], format: "2017-06-23 15:19:02"
##   ..$ ThetaV   : num 35
##   ..$ Dphi     : num 135
##   ..$ Windspeed: num 12
##   ..$ ThetaS   : num 29.9
##   ..$ PhiS     : num 147
##  $ CLEARSKY        : logi FALSE
```

La fonction `plot.ASD.rhow()` permet de visualiser les spectres de réflectance marine calculés selon la méthode décrite ci-haut et avec ou sans la correction pour contamination résiduelle, ϵ, causée par l'environnement.

```
plot.ASD.rhow(rhow)
```

Dans cet exemple, on peut voir que la correction est importante. Celle qui emploie λ_1=720 et λ_2=780 donne un spetre fortement négatif dans le PIR. Les corrections qui supposent que la réflectance est égale à 0 à 900 nm et celle qui utilise λ_1=780 et λ_2=870 donnent des résultats relativement similaires. Cependant la forme de ρ_w dans l'UV (remonter à mesure de λ diminue) avec ces trois méthodes est douteuse dans les eaux riches en CDOM. Dans cette exemple, ce sont les méthodes qui emploient le pixel noir dans le PIR ou l'UV pour estimer le $\rho_{Fresnel}$ qui donnent les meilleurs résultats. Les valeurs de $\rho_{Fresnel}$ sont pratiquement identiques (~0.41), mais nettement supérieur à la valeur proposée par Ruddick et al (2006) pour une ciel couvert (0.0256) (voir aussi Mobley, 1999 pour les valeurs possibles pour des ciels nuageux.).

On peut également visualiser les données brutes avec la même fonction mais en mettant l'argument RADIANCES=TRUE.

```
plot.ASD.rhow(rhow, RADIANCES = TRUE)
```

Cette figure montre la variatilité des luminances mesurées (zone grisée) pour chaque type de mesure et la réflectance de la surface (tireté) et du ciel rélféchi vers le capteur (continu). On voit bien que l'augmentation de la réflectnace dans l'UV vient de la réflectance du ciel à l'interface.

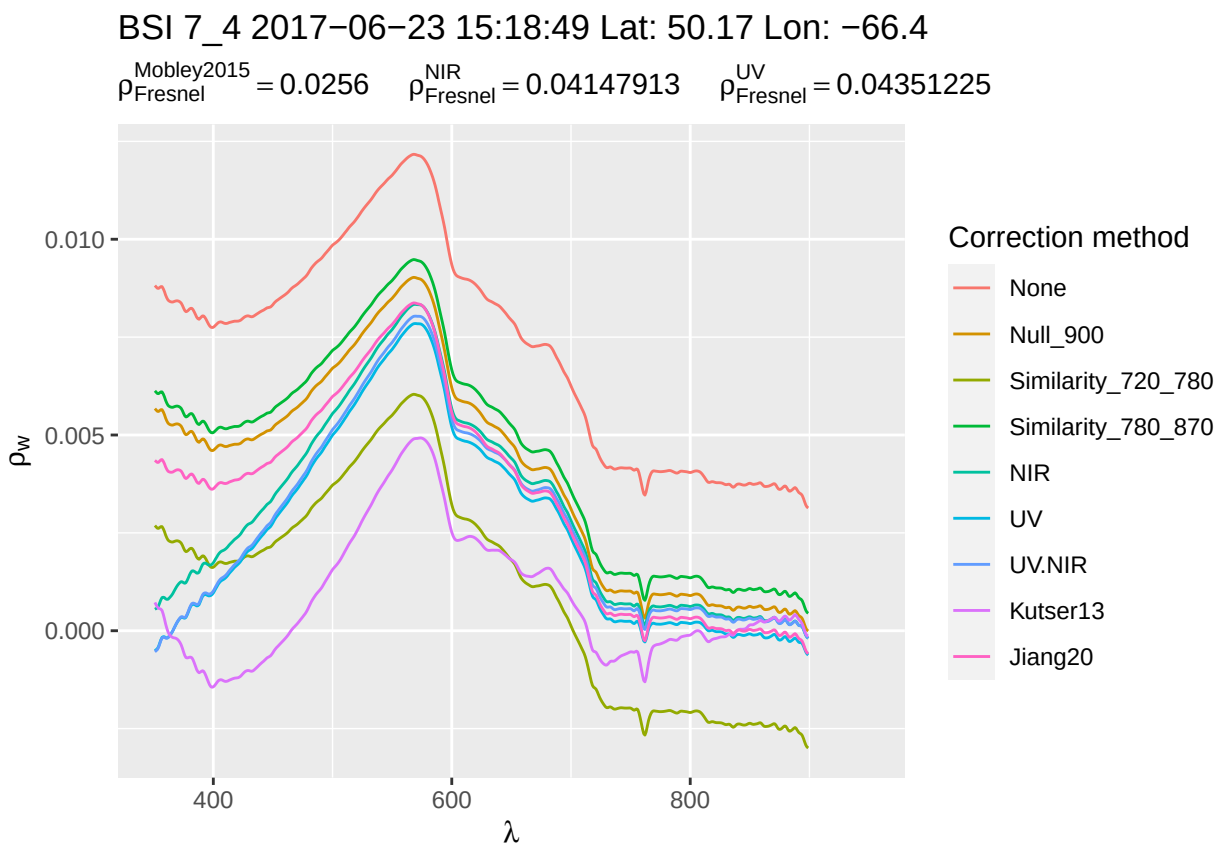


Figure 2: Réflectance marine pour la Station BSI 7_4 avec l'option quantile.prod=0.75

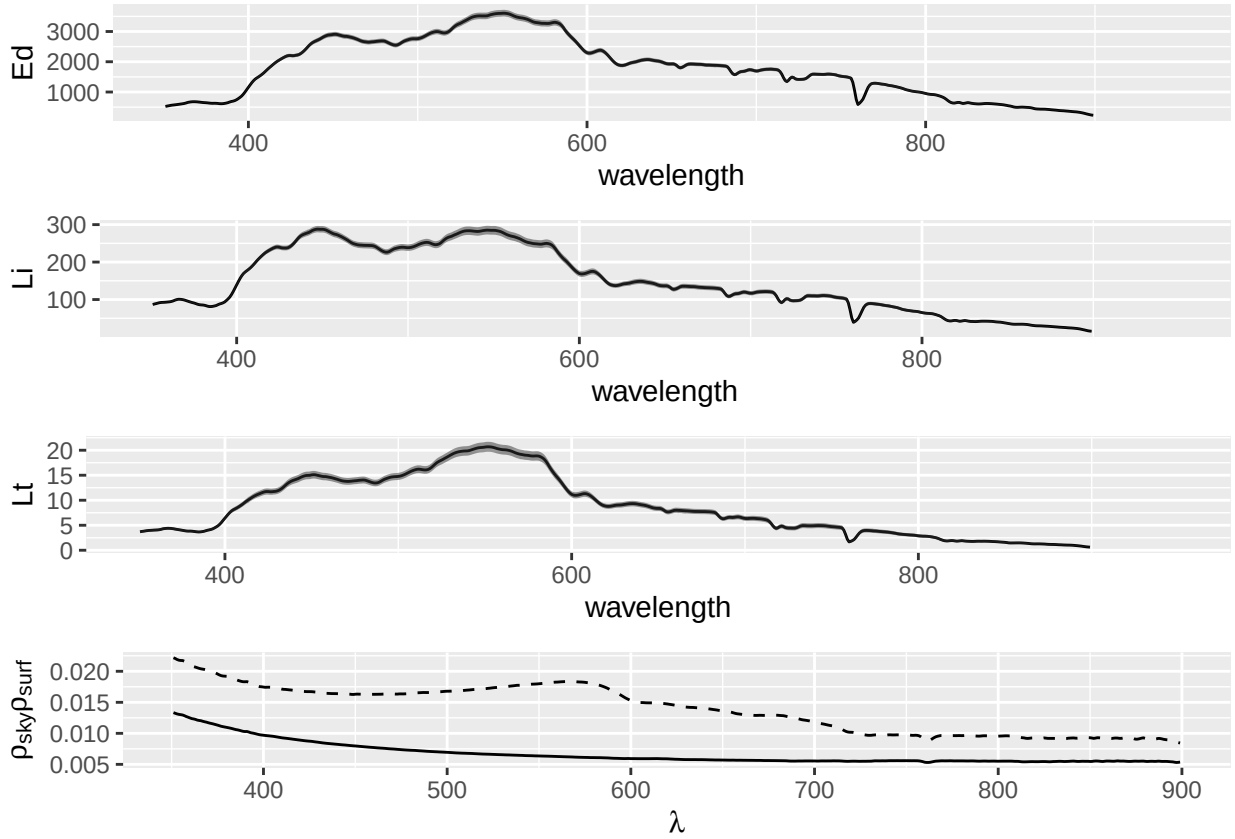


Figure 3: Données brutes de la BSI 7_4. La ligne tiretée sur la dernière figure est la réflectance de la surface totale et la ligne continue la réflectance du ciel calculé avec le $rho_{Fresnel}$ de la table de Mobley

Une telle inspection des spectres devrait être faites a posteriori et la meilleure méthode sélectionnée.

```
rhow = compute.ASD.rhow(ASDtot, 0.985, quantile.prob = 0.5)
```

```
## [1] "Averaging Radiance spcetra..."
## [1] "Compute Sky and Surface reflectance and apply a smoothing function"
## [1] "Cloudy sky, Use rho = 0.0256"
## [1] "Apply NIR corrections"
## [1] "Begining the Kutser correction for glint"
## [1] "Wavelength and data at UV and NIR binned"
## [1] "Starting the NLS"
## [1] "Kutser correction finished"
## List of 31
##  $ waves           : int [1:751] 325 326 327 328 329 330 331 332 333 334 ...
##  $ rhow            : num [1:751] 0.00927 0.00948 0.00954 0.00946 0.00931 ...
##  $ rhow.NULL       : num [1:751] 0.00655 0.00676 0.00682 0.00675 0.00659 ...
##  $ rhow.SIMILARITY1: num [1:751] 0.00384 0.00405 0.00411 0.00404 0.00388 ...
##  $ rhow.SIMILARITY2: num [1:751] 0.00693 0.00714 0.0072 0.00712 0.00697 ...
##  $ rhow.NIR        : num [1:751] 0.000603 0.000852 0.000957 0.000928 0.000824 ...
##  $ rhow.UV         : num [1:751] -0.00142 -0.00117 -0.00105 -0.00107 -0.00116 ...
##  $ rhow.UV.NIR     : num [1:751] -0.00158 -0.00131 -0.00119 -0.00121 -0.0013 ...
##  $ rhow.COPS       : num NA
##  $ rhow.Kutser     : num [1:751] 0.00258 0.00279 0.00284 0.00274 0.00257 ...
##  $ rhow.Jiang      : num [1:751] 0.00498 0.0052 0.00526 0.0052 0.00505 ...
##  $ rho.sky         : num 0.0256
##  $ rho.sky.NIR     : num 0.0394
##  $ rho.sky.UV      : num 0.0427
##  $ rho.sky.UV.NIR  : num [1:751] 0.0429 0.0429 0.0429 0.0429 0.0429 ...
##  $ rho.sky.COPS    : logi NA
##  $ Lpanel          : num [1:751, 1:16] 18.8 24.2 30.9 38.4 46.6 ...
##  $ Lt              : num [1:751, 1:24] 0.533 0.69 0.852 1.039 1.249 ...
##  $ Li              : num [1:751, 1:16] 12.8 16.6 20.6 25.1 30.1 ...
##  $ Ed.mean         : num [1:751] 64.7 82.1 101.6 126 154.7 ...
##  $ Ed.sd           : num [1:751] 1.91 2.22 1.92 2.16 2.49 ...
##  $ Lt.mean         : num [1:751] 0.517 0.669 0.833 1.02 1.226 ...
##  $ Lt.sd           : num [1:751] 0.0263 0.0355 0.0455 0.0559 0.0662 ...
##  $ Li.mean         : num [1:751] 12.7 16.3 20.5 25.1 30 ...
##  $ Li.sd           : num [1:751] 0.148 0.203 0.198 0.23 0.28 ...
##  $ ix.Lt.good      : int [1:24] 5 6 8 9 10 11 12 14 15 16 ...
##  $ ix.Lsky.good    : int [1:16] 1 4 5 6 7 8 9 10 11 12 ...
##  $ ix.Lpanel.good  : int [1:16] 3 4 5 6 7 8 9 10 11 12 ...
##  $ DateTime        : POSIXct[1:1], format: "2017-06-23 15:18:46"
##  $ anc             :List of 9
##   ..$ StationID: chr "BSI 7_4"
##   ..$ lat      : num 50.2
##   ..$ lon      : num -66.4
##   ..$ DateTime : POSIXct[1:1], format: "2017-06-23 15:19:02"
##   ..$ ThetaV   : num 35
##   ..$ Dphi     : num 135
##   ..$ Windspeed: num 12
##   ..$ ThetaS   : num 29.9
##   ..$ PhiS     : num 147
##  $ CLEARSKY        : logi FALSE
```

```
plot.ASD.rhow(rhow)
```

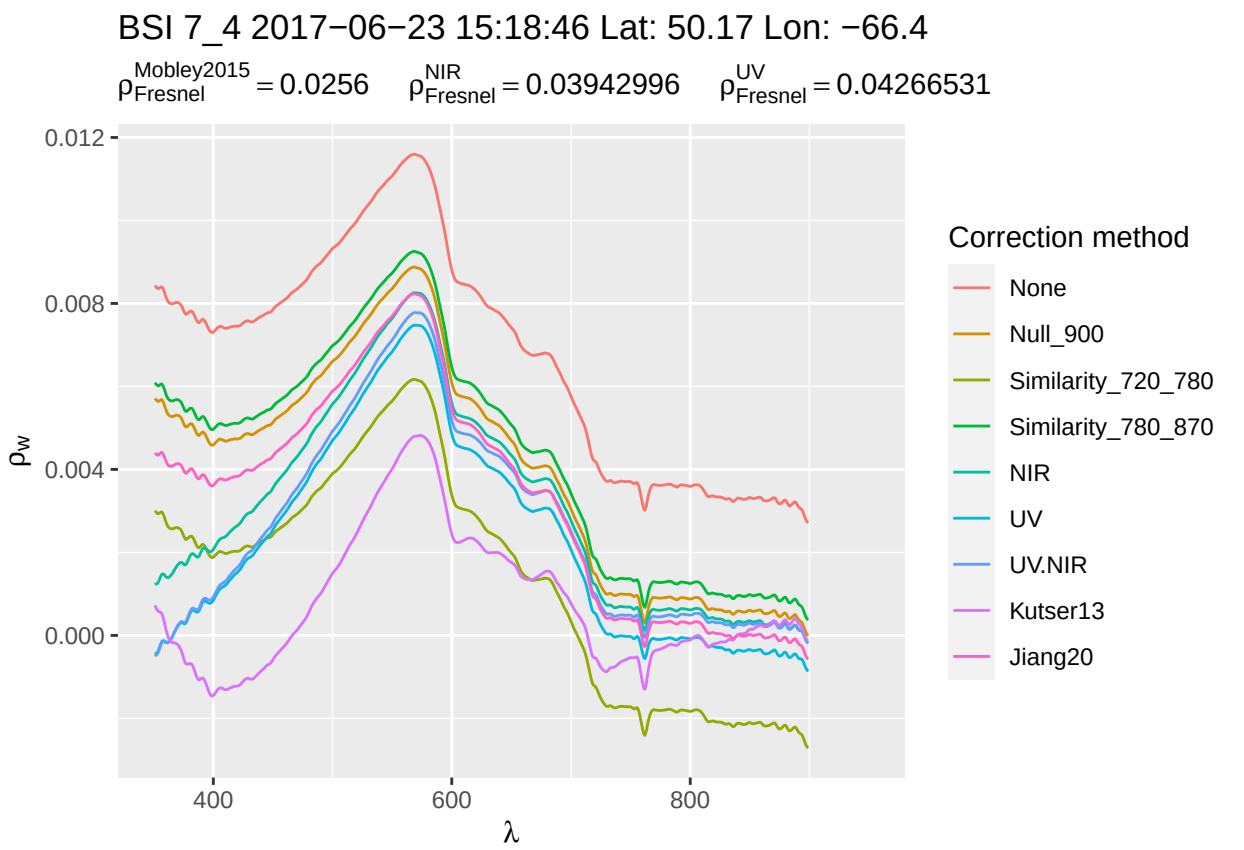


Figure 4: Réflectance marine pour la Station BSI 7_4 avec l'option quantile.prod=0.5

On voit peu de différence entre *quantile.prob* = 0.75 et *quantile.prob* = 0.5 dans cette exemple bien que de le $\rho_{Fresnel}^{BP}$ soit plus faible avec *quantile.prob* = 0.5 car on a éliminé plus de forte valeur de Lt dans le calcul de la moyenne.

Procédure pour le traitement semi-automatisé des surfaces aquatiques

Afin de simplifier le traitement des données, l'utilisateur pourra utiliser la fonction `ASD.go()`. Cette fonction sera lancée à partir d'un **répertoire de travail** qui sera défini au préalable avec la commande `setwd()` de R. Mais auparavant, il est important de comprendre comment organiser les données et les métadonnées dans une struture de répertoire logique.

Étape 1 : préparation des données

Tout d'abord, les fichiers ASCII de l'asd (.asd.txt) doivent être placés dans un répertoire ou plusieurs répertoires. Ces répertoires seront nos **répertoires de données**. On peut par exemple créer un répartoire par station et y mettre les fichiers ASD pris à cette station.

Étape 2 : préparation des fichiers ASCII pour le traitement des données

Le traitement des données est lancé par la fonction `ASD.go()` qui lira deux type de fichiers texte (ASCII).

- Le **répertoire de travail** d'où sera lancé `ASD.go()` doit absolument contenir un fichier nommé **directories.for.ASD.dat**. Ce dernier contient les chemins des **répertoires de données** à traiter. Il est ainsi possible de lancer le traitement sur un ou plusieurs répertoires à la fois.

- Dans chaque **répertoires de données** à traiter, il faut mettre un fichier nommé **cast.info.dat** qui est un fichier texte où chaque champ est séparé d'un espace. Si le fichier est absent d'un **répertoire de données**, un fichier **cast.info.dat** sera automatiquement créé dans le répertoire. Il sera important d'éditer manuellement ce fichier. Le fichier **cast.info.dat** comprend 16 colonnes et autant de lignes que de spectres marine qu'on veut calculer. Les 16 colonnes sont décrites en détail à la section *Acquisition des données ASD sur le terrain* (p. ex:, lat lon basename ID ...).

Étape 3 : lancement du traitement automatisé

Quand notre fichier **directories.for.ASD.dat** est créé et que chaque **répertoire de données** contient un fichier **cast.info.dat**, on peut lancer le code avec `ASD.go()`. Cette fonction accepte deux arguments logiques:

- **PNG** indique que les figures produites par `plot.ASD.rhow` seront sauvegardées en format png dans un sous-répertoire nommé PNG. Les PNG sont nommés avec le champs "ID" trouvé dans **cast.info.dat**. Par défault PNG = FALSE et donc les figures ne sont pas sauvegardées en png.
- **ADD_UNDERSCORE** indique que les noms des fichiers ASD contiennent un " *" entre la base et les numéros de fichier. Habituellement les fichiers sont nommés "StationX_00001.asd.txt" où la base "StationX" est séparée des numéros par un "*". Ainsi par défaut ADD_UNDERSCORE = TRUE

Une fois lancé, chaque répertoire sera traité séquentiellement et chaque "cast" trouvé dans **cast.info.dat** sera traité. Pour chaque cast, un fichier RData sera créé et nommé avec le champs "ID" trouvé dans **cast.info.dat**. Les fichiers RData contiennent l'objet rhow. Il y aura autant de fichiers RData créés que de casts traités.

Exemple

Dans l'exemple qui suit, on a mis 110 fichiers ASD dans le répertoire "~/MEGA/data/MicroCASI_project/L2/20170911_Station
Deux surfaces avaient été mesursées à cette station dans la baie de Laval en eaux peu profonde où on avait un fond sableux à proximité d'un amat de laminaires. Ainsi le fichier cast.info.dat contient deux lignes contenant les 16 champs nécessaire pour le traitement.

```
##        lat      lon  basename              ID Lpanel_start Lpanel_end
## 1 48.75746 -69.0344 baielaval     BL-01_sable            0         19
## 2 48.75746 -69.0344 baielaval BL-01_laminaire            0         19
##   Lsky_start Lsky_end Ltot_start Ltot_end ThetaV Dphi Windspeed Wind.units
## 1         20       39         40       69     40  135        12        Kts
## 2         20       39         70      109     35  135        12        Kts
##   quantile.prob rhow.Method
## 1           0.7           0
## 2           0.7           0
```

On remarque qu'on a utlisé les mêmes mesures pour éclairement et le ciel pour nos deux surfaces puisqu'elles ont été mesurées à quelques secondes d'intervalle.

```
library(asdsvc)
setwd("~/OneDrive - UQAR/data/MicroCASI_project/L2/20170911_StationBL-01/ASD/")
ASD.go(PNG=TRUE)
```

Le traitement a produit deux fichiers RData et 4 fichiers PNG.

```
##  [1] "BL-01_laminaire_Radiances.png"  "BL-01_laminaire_rhow.png"
##  [3] "BL-01_laminaire.ASD.rhow.RData" "BL-01_sable_Radiances.png"
##  [5] "BL-01_sable_rhow.png"           "BL-01_sable.ASD.rhow.RData"
##  [7] "BL01_laminaire_Radiances.png"   "BL01_laminaire_rhow.png"
##  [9] "BL01_laminaire.ASD.rhow.RData"  "BL01_sable_Radiances.png"
## [11] "BL01_sable_rhow.png"            "BL01_sable.ASD.rhow.RData"
```

Calcul de la réflectance spectrale pour les surfaces terrestres

On calcule la réflectance pour les surfaces terrestes se fait avec la fonction `compute.ASD.rho()`. Cette fonction a été développé pour le cours de télédétection. Elle est plus simple à utiliser car elle ne requiert pas les données auxiliaires comme pour la réfelctance de l'eau.

Entre les parenthèses, on doit fournir trois arguments à `compute.ASD.rho()`:

- le premier argument est le nom du fichier qui contient la luminance du spectralon;
- le deuxième argument est le nom du fichier qui contient la luminance de la surface qu'on veut traiter;
- le dernier est la réflectance du panneau de spectralon (par défaut est 0.98, soit 98%), cet argument peut être omis.

Suposons qu'on a mesuré une surface herbacée avec l'ASD ("20160920_Site400003.asd.txt"). Avant nous avions mesuré le spectralon afin d'estimer l'éclairement incident nécessaire pour calculer la réflectance ("20160920_Site400000.asd.txt").

Exemple:

```
#setwd("~/Google Drive/Cours/IntroTeledec/SortieTerrain/DonneesASD/")
setwd("~/OneDrive - UQAR/Cours/IntroTeledec/SortieTerrain/DonneesASD/")
ReflecHerbace <- compute.ASD.rho("20160920_Site400000.asd.txt", "20160920_Site400003.asd.txt", 0.985)
```

```
## [1] "20160920_Site400000.asd.txt"
## [1] "20160920_Site400003.asd.txt"
```

La fonction `compute.ASD.rho()` a retourné un objet qui contient la réflectance qui a été nommé **ReflecHerbace**. Vous pouvez renommer cet objet si vous désirez (préférable). Il faudra cependant penser à le changer dans le reste du script.

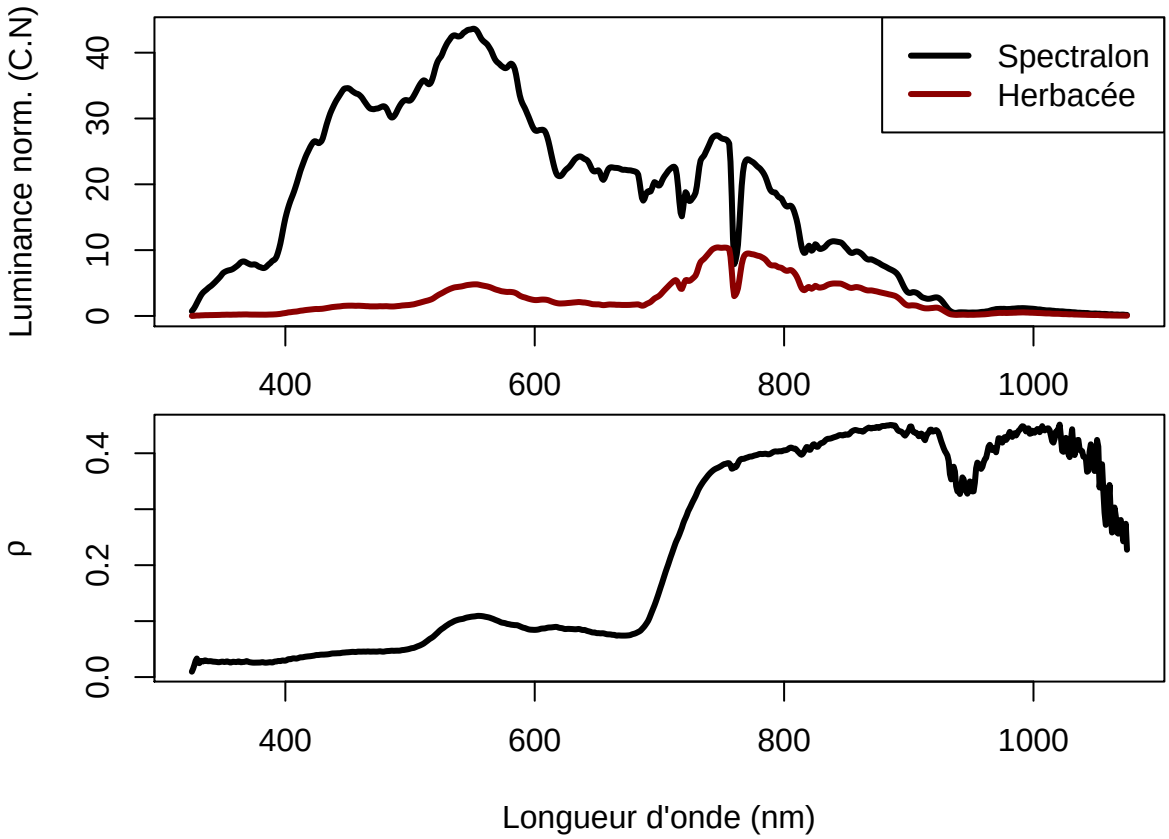
Visualiser les spectres de réflectance

La fonction `plot.ASD.rho()` permet de faire une figure avec les données de l'objet retourné par la fonction `compute.ASD.rho()`. La fonction `plot.ASD.rho()` comprend trois arguments:

- le premier argument est l'objet qui a été créé par la fonction `compute.ASD.rho()` (**ReflecHerbace** dans notre exemple);
- le deuxième argument est une chaine de caractères entre guillets "" qui correspond au nom de la surface qui apparaitra dans la légende de la figure.
- Le troisième argument se nomme PNG est un argument logique qui prend la valeur TRUE ou FALSE. PNG = TRUE indique qu'une figure en format png sera sauvegarder dans le répertoire de travail.

Exemple:

```
plot.ASD.rho(ReflecHerbace, "Herbacée", PNG=FALSE)
```




```
# pour créer le fichier png:
plot.ASD.rho(ReflecHerbace, "Herbacée", PNG=TRUE)
```

La figure comprend deux paneaux superposés. Celui du haut montre les spectres de luminance qui ont été normalisés par le temps d'intégration de la mesure ASD. Celui du bas est la réflectance calculée.

Exporter les données de réflectance

On peut aussi écrire le résultat dans un fichier texte lisible dans un autre logiciel (type ASCII). On pourrait vouloir travailler avec EXCEL pour combiner des spectres sur une même figure. La fonction `export.ASD.rho()` accepte deux arguments:

- le premier argument est l'objet qui a été créé par la fonction `compute.ASD.rho()` (**ReflecHerbace** dans notre exemple);
- le deuxième est le nom du fichier entre guillemets.

Exemple:

```
export.ASD.rho(ReflecHerbace, "Herbacée.txt")
```

Le fichier "Herbacée.txt" comprend deux colonnes, soit la longueur d'onde en nanomètre (wavelenght) et la réflectance ($\rho = \pi L / E_d$).

References

Mobley, Curtis D. 1999. "Estimation of the remote-sensing réflectances from above-water measurements." *Applied Optics* 38 (36): 7442–55.

———. 2015. “Polarized reflectance and transmittance properties of windblown sea surfaces.” *Appl. Opt.* 54 (15): 4828–49. https://doi.org/10.1364/AO.54.004828.

Ruddick, Kevin, Vera De Cauwer, and Barbara Van Mol. 2005. “Use of the near infrared similarity reflectance spectrum for the quality control of remote sensing data” 2005 (August).

Ruddick, Kevin G, Vera De Cauwer, Young-Je Park, and Gerald Moore. 2006. “Seaborne measurements of near infrared water-leaving reflectance: The similarity spectrum for turbid waters.” *Limnol. Oceanogr.* 51 (2): 1167–79.