MEDSLIK-II version 2.01

User Manual

15 June 2020

Title: MEDSLIK-II v2.01 User Manual	
Reference: http://medslik-ii.org/users/account.php Dissemination: Free (online)	Date: 2020/June/15 Version: Final Language: English
Authors: S. Liubartseva (CMCC, Bologna, Italy), F. Trotta (UNIBO, Bologna, Italy), N. Pinardi (UNIBO, Bologna, Italy), F. Viola (CMCC, Lecce, Italy), M. Scuro (CMCC, Lecce, Italy), S. Cretí (CMCC, Lecce, Italy), G. Coppini (CMCC, Lecce, Italy), R. Lecci (CMCC, Lecce, Italy), P. Agostini (CMCC, Lecce, Italy)	
Key words: Oil spill pollution; Lagrangian modeling; Global Ocean	
How to cite: Liubartseva, S., Trotta, F., Pinardi, N., Viola, F., Scuro, M., Cretí, S., Coppini, G., Lecci, R., & Agostini, P. (2020). MEDSLIK_II v2.01. Centro Euro-Mediterraneo Sui Cambiamenti Climatici. https://doi.org/10.25423/CMCC/MEDSLIK_II_2.01_USER_MANUAL	
DOI: https://doi.org/10.25423/CMCC/MEDSLIK_II_2.01_USER_MANUAL	

Table of content

Change Record

Date	Author	Affiliation	Description of changes	Version	Status
18.02.2020	Svitlana Liubartseva	CMCC	Self-contained Manual based on the previous MEDSLIK-II 1.01, 1.02, 2.0 Manual versions and the latest model update	1	Draft
09.03.2020	Svitlana Liubartseva, Nadia Pinardi	CMCC, UNIBO	Re-structured package	2	Draft
18.03.2020	Svitlana Liubartseva, Nadia Pinardi	CMCC, UNIBO	Review of the text, renaming the files	3	Draft
20.04.2020	Svitlana Liubartseva, Nadia Pinardi	CMCC, UNIBO	Review of the text, reducing the simulation length for the *Paria* test case, providing the Lebanon test case	4	Draft
22.05.2020	Francesco Trotta	UNIBO	First overall revision	5	Draft
25.05.2020	Fabio Viola	CMCC	Porting Python scripts to Python 3 Implementing `conf.json`	6	Draft
03.06.2020	Francesco Trotta Fabio Viola Svitlana Liubartseva	UNIBO CMCC CMCC	Next overall revision Further development of the Python scripts and `conf.json`, update of the documentation Update of the documentation	7	Draft
08.06.2020	Francesco Trotta Svitlana Liubartseva	UNIBO CMCC	Final overall revision Update of the documentation	8	Draft
12.06.2020	Francesco Trotta Nadia Pinardi	UNIBO	Writing the Section ‘Software requirements’ in a user-friendly style	9	Draft
15.06.2020	Nadia Pinardi	UNIBO	Last corrections to the English	10	Final

1. Introduction

MEDSLIK-II was originally conceived to carry out simulations supporting oil spill emergency situations. The oil spill forecasting has evolved, and now oil spill modelling uncertainties are evaluated by ensemble methods (Sepp-Neves et al., 2016).
MEDSLIK-II v2.01 inherited the MEDSLIK-II v2.0 improvements related to the application of the model to the global ocean, the outputs with a netCDF format. This Manual is aimed at describing the MEDSLIK-II v2.01 version (available at the model website) in a self-contained way. Thus, it describes the actual model features and the features inherited from the previous versions and previously published in the Manuals at the model website (De Dominicis, 2012; Bruciaferri et al., 2015; Sepp-Neves, 2018).
This manual, as well as the previous ones and the codes are all available from the perpetual web page: http://www.medslik-ii.org/

This Manual is organized as follows:

- In Section 2, a summary of the new features of the version MEDSLIK-II v2.01 is given.
- In Section 3, the governing equations solved by the model and the physical processes involved are briefly described. Additionally, the model solution methodology is presented.
- In Section 4, the software installation procedure and the description of the code structure are given.
- In Section 5, the steps needed to run oil spill simulation are described in detail.
- In Section 6, the bug fixing and upgrade of the Python scripts are described.
- In Section 7, the *Paria* test case performed is described to validate the model.
- In Section 8, the Lebanon test case is provided to test the model additionally.

2. MEDSLIK-II v2.01

The modified code, hereinafter MEDSLIK- II v2.01, inherits the main features from MEDSLIK- II v2.0, as follows:

- Ingestion of global bathymetry from GEBCO (https://www.gebco.net) database with original 30" resolution and coastline based on the NOAA GSHHG data set (https://www.ngdc.noaa.gov/mgg/shorelines/data/gshhg/latest/);
- Ingestion of global meteo-oceanographic inputs from the ERA-Interim atmospheric fields (https://apps.ecmwf.int/datasets/data/interim-full-daily/levtype=sfc/) and GLOBAL OCEAN 1/12° PHYSICS ANALYSIS AND FORECAST (http://marine.copernicus.eu/services-portfolio/access-to-products/?option=com_csw&view=details&product_id=GLOBAL_ANALYSIS_FORECAST_PHY_001_024) provided by CMEMS;
- Generating netCDF4 outputs

In comparison with MEDSLIK- II v2.0, MEDSLIK- II v2.01 version includes bug fixes. The Python scripts are ported from Python 2 to Python 3.

Additionally, a configuration file `conf.json` is implemented. This file is used to set model parameters and it needs to be passed as an argument to the python scripts.

3. Governing Equations and Model Methodology

The oil spill model theoretical and numerical framework can be found in De Dominicis et al. (2013a) and De Dominicis et al. (2013b). Here we will overview only some key points useful to start using the model.
MEDSLIK-II is designed to simulate oil slick transport and transformation processes from a surface oil release (later versions could contain subsurface oil sources, but at the moment, the release is always assumed at the surface). In the following, we overview the partial differential equations and the finite difference schemes used to predict the oil slick changes at sea and its transport.

3.1 Model state variables and governing equations

The general equation for a tracer concentration, $C(\boldsymbol{x}, t)$, with units of mass over volume, mixed in the marine environment, is

$$\frac{\partial C}{\partial t} + \boldsymbol{U}\boldsymbol{\nabla} C = \boldsymbol{\nabla}(\boldsymbol{K}\boldsymbol{\nabla} C) + \sum_{j=1}^{M} r_j\left(\boldsymbol{x}, C(\boldsymbol{x}, t), t\right), \tag{1}$$

where $\frac{\partial}{\partial t}$ is the local time-rate-of-change operator, $\boldsymbol{U}$ is a three-dimensional distribution of the horizontal ocean current components $\boldsymbol{u}$ and $\boldsymbol{v}$, $\boldsymbol{K}$ is a turbulent diffusivity tensor, which parameterizes the sub-grid scale processes; and $r_j(\boldsymbol{x}, C(\boldsymbol{x}, t), t)$ are the M transformation rates that modify the tracer concentration by means of physical and chemical transformation processes.
Assuming that the slick is composed of oil constituent particles, which move like water parcels, and considering that the physical and chemical processes act on the entire slick rather than on the single particle properties, the active tracer equation can be effectively split into two component equations:

$$\frac{\partial C_1}{\partial t} = \sum_{j=1}^{M} r_j\left(\boldsymbol{x}, C_1(\boldsymbol{x}, t), t\right) \tag{2}$$

$$\frac{\partial C}{\partial t} = -\boldsymbol{U}\boldsymbol{\nabla} C_1 + \boldsymbol{\nabla}(\boldsymbol{K}\boldsymbol{\nabla} C_1) \tag{3}$$

where C_1 is the oil concentration solution solely due to the weathering processes, while the final time rate of change of C is given by the advection-diffusion acting on C_1.
The second equation describes the evolution in time of the oil slick concentration due to the transformation processes acting on the total oil slick volume: therefore, oil slick state variables have to be defined.
On the other hand, in order to solve the governing equation by using a Lagrangian particle formalism, it is needed to discretize the oil slick in particles with associated particle state variables, some of them deduced from the oil slick state variables.
Finally, four structural state variables C_S, C_D, C_C, and C_B are defined, which describes, respectively, the oil concentration at the surface, in the sub-surface, adsorbed on the coasts and sedimented at the bottom (see Fig. 1).

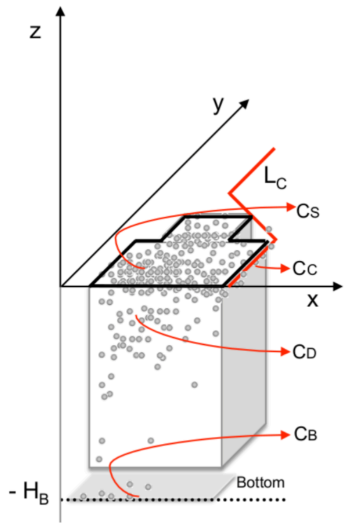


Fig. 1 Schematic representation of the oil concentration classes (the grey spheres represent the oil particles and L_C – the coastline) (De Dominicis et al., 2013a).

Table 1 Structural, slick and particle state variables defined in the model (De Dominicis et al., 2013a)

Variable	**Variable type**	**Description**	**Units**
$C_S(\boldsymbol{x}, t)$	Structural	Oil concentration at the sea surface	kg m^{-2}
$C_D(\boldsymbol{x}, t)$	Structural	Oil concentration of dispersed fraction	kg m^{-2}
$C_C(\boldsymbol{x}, t)$	Structural	Oil concentration on the coast	kg m^{-2}
$C_B(\boldsymbol{x}, t)$	Structural	Oil concentration on the bottom	kg m^{-2}
$V_S(\boldsymbol{x}, t)$	Slick	Surface oil slick volume	m^3
$V_D(\boldsymbol{x}, t)$	Slick	Dispersed oil slick volume	m^3
$V^{TK}(\boldsymbol{x}, t)$	Slick	Thick part of the surface oil slick volume	m^3
$V^{TN}(\boldsymbol{x}, t)$	Slick	Thin part of the surface oil slick volume	m^3
$A^{TK}(\boldsymbol{x}, t)$	Slick	Thick part area of the surface oil slick	m^2
$A^{TN}(\boldsymbol{x}, t)$	Slick	Thin part area of the surface oil slick	m^2
$T^{TK}(\boldsymbol{x}, t)$	Slick	Thick part thickness of the surface oil slick	m
$T^{TN}(\boldsymbol{x}, t)$	Slick	Thin part thickness of the surface oil slick	m
$\boldsymbol{x}_k(t) = (x_k(t), y_k(t), z_k(t))$	Particle	Particle coordinate	m
$v_E(k, t)$	Particle	Evaporative surface oil volume	m^3
$v_{NE}(k, t)$	Particle	Non-evaporative surface oil volume	m^3
$v_{NE}(k, t)$			
$\sigma(k, t) = 0, 1, 2, <0$	Particle	Particle status index (at surface, dispersed, on bottom, on coast)	n/a

In the model, the concentration C_1 represents the oil concentration at the surface C_S and the oil concentration dispersed in the water column C_D. Therefore, the oil weathering equation comprehends the following

$$\frac{dC_S}{dt} = \frac{\rho}{A}\frac{dV_S}{dt} \tag{4}$$

$$\frac{dC_D}{dt} = \frac{\rho}{A}\frac{dV_D}{dt} \tag{5},$$

where A is the unit area, V_S is the surface oil slick volume, V_D is the dispersed oil volume and ρ is the sea water density.
Regarding the surface oil which arrives close to the coasts and is adsorbed, the concentration of beached oil is given by

$$C_C(x,t) = \frac{\rho}{L_C} V_C, \tag{6}$$

where L_C is a coastline segment and V_C is the adsorbed oil volume calculated from the oil particle state variables.
In the present version of the model, the oil concentration on the bottom, C_B is not computed, and it is simply represented by a number of oil particles that reached the bottom.
The weathering acting on the sea surface oil comprehends three main processes: evaporation, which occurs for first and mainly acts on the lighter fractions of the oil, dispersion of the remaining oil fractions below the water surface and spreading, which is responsible for the mechanical spreading of the spill over the water surface under the action of gravitational forces (Fig. 2).

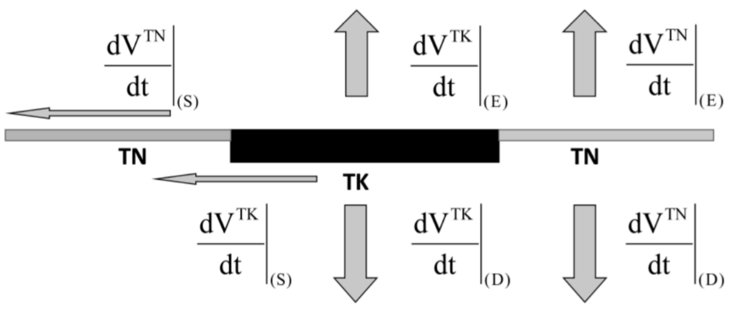

$\left.\frac{dV^{TN}}{dt}\right|_{(S)}$
$\left.\frac{dV^{TK}}{dt}\right|_{(E)}$
$\left.\frac{dV^{TN}}{dt}\right|_{(E)}$
TN
TK
TN
$\left.\frac{dV^{TK}}{dt}\right|_{(S)}$
$\left.\frac{dV^{TK}}{dt}\right|_{(D)}$
$\left.\frac{dV^{TN}}{dt}\right|_{(D)}$

Fig. 2 Weathering processes using Mackay's approach. *TK* indicates the thick slick and *TN* the thin slick. V^{TK} and V^{TN} are the surface oil volumes of the thick and thin part of the slick and the suffixes indicate evaporation (*E*), dispersion (*D*) and spreading (*S*) (De Dominicis et al., 2013a).

Consequently, the weathering processes are considered separately for the thick slick and thin slick and the prognostic equations are

$$\frac{dV^{TK}}{dt} = \left.\frac{dV^{TK}}{dt}\right|(E) + \left.\frac{dV^{TK}}{dt}\right|(D) + \left.\frac{dV^{TK}}{dt}\right|(S) \tag{7}$$

$$\frac{dV^{TN}}{dt} = \left.\frac{dV^{TN}}{dt}\right|(E) + \left.\frac{dV^{TN}}{dt}\right|(D) + \left.\frac{dV^{TN}}{dt}\right|(S) \tag{8},$$

where the suffixes indicate evaporation (*E*), dispersion (*D*) and spreading (*S*).
All the oil slick state variables are defined only at the slick's central geographical position (Gravity Centre, hereinafter *GC*), which is updated after each time step. Parameterization of the weathering processes is detailed in De Dominicis et al. (2013a). Initially, the model solves the weathering equation, and then uses the oil concentration solution C_1 to solve the advection-diffusion equation.

In order to solve the latter with a Lagrangian particle formalism, the oil slick is initially discretized in N_{TOT} particles with associated particle state variables and then the oil concentration is computed by assembling the particles together with their associated properties.
The horizontal current field used in the Lagrangian model is taken to be the sum of different components:

$$\begin{cases} \sigma = 0, & d\boldsymbol{x}_k(t) = [\boldsymbol{U}_C(x_k, y_k, 0, t) + \boldsymbol{U}_W(x_k, y_k, 0, t) + \\ & +\boldsymbol{U}_S(x_k, y_k, 0, t) + \boldsymbol{U}_D(x_k, y_k, 0, t)]dt + d\boldsymbol{x'}_k(t), \\ \sigma = 1, & d\boldsymbol{x}_k(t) = \boldsymbol{U}_C(x_k, y_k, z_k, t)dt + d\boldsymbol{x'}_k(t) \end{cases} \tag{9}$$

where dt is the discrete model time step, $\boldsymbol{x}$=(x, y) is a horizontal position vector; σ is the particle status index which identifies if a particle is on the sea surface ($\sigma = 0$) or dispersed into the water column ($\sigma = 1$), U_S is the ocean current velocity field, U_S is the local wind velocity correction term (wind drift term), U_S called hereafter wave current term, is the velocity due to wave-induced currents or the surface Stokes drift, U_S is the wind drag correction due to emergent part of the objects at the surface and $d\boldsymbol{x}'_k$ is the displacement due to the Lagrangian turbulent diffusion. The parcel becomes dispersed only for transformation processes and not for vertical diffusion or advection.
In the model, the turbulent diffusion is considered at the moment only for Brownian motion parametrized by a random walk scheme as follows:

$$d\boldsymbol{x'}_k(t) = \sqrt{6\boldsymbol{K}dt}\boldsymbol{Z}, \tag{10}$$

where $\boldsymbol{K}$ is the turbulent diffusion diagonal tensor and $\boldsymbol{Z}$ is a vector of independent random numbers distributed uniformly. Usually, $\boldsymbol{K}$ is taken to be isotropic and equal to 2 m^2s^{-1}.

3.2 Sequential solution methodology

In order to solve coherently the weathering and the advection-diffusion equations, an algorithm, which links the oil slick and the particle state variables is needed. The model sequential solution method is represented schematically in Fig. 3.
A surface oil release into the marine environment can be instantaneous or continuous: in the first case, all the oil is spilled instantly, while in the latter, the leakage may last for several hours or even months. Furthermore, the simulated scenario could involve an oil slick detected by satellite or at sight.
The continuous oil spill release is modeled dividing the total spilled oil into a number of sub-spills consisting of a given part of the oil released at the spill location during the prescribed computational time interval. As each sub-spill is moved away from the source, the total spill becomes a chain of sub-spills. In the continuous release, a given fraction of the total amount of particles used in the simulation is released with each sub-spill.

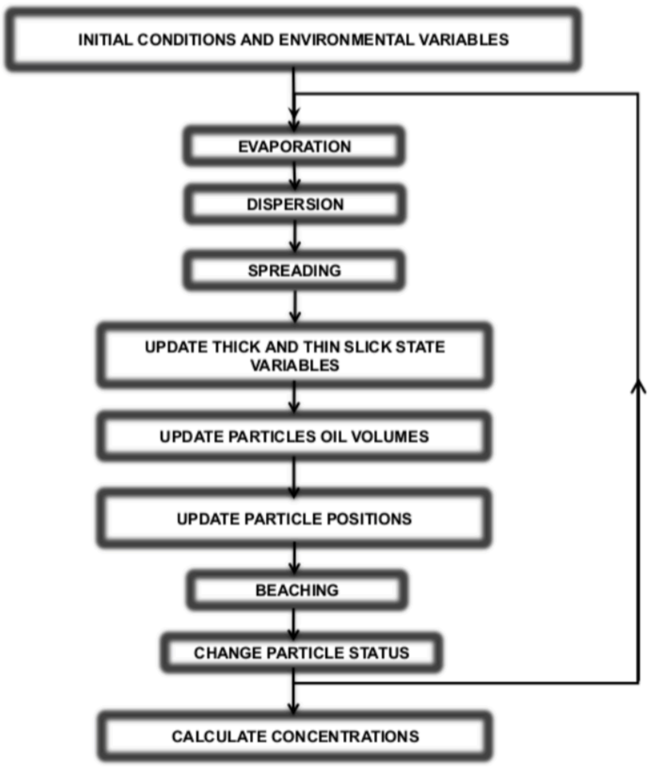

INITIAL CONDITIONS AND ENVIRONMENTAL VARIABLES
EVAPORATION
DISPERSION
SPREADING
UPDATE THICK AND THIN SLICK STATE VARIABLES
UPDATE PARTICLES OIL VOLUMES
UPDATE PARTICLE POSITIONS
BEACHING
CHANGE PARTICLE STATUS
CALCULATE CONCENTRATIONS

Fig. 3 Model sequential solution method (taken from De Dominicis et al., 2013a).

In the case of an instantaneous release, at the beginning of the simulation, all the oil is released on the sea surface at the spill location and all the particles are released instantaneously.
Satellite or at sight detected slicks are modeled as instantaneous spills except for the fact that the total amount of oil and particles of each slick are released inside the polygon which identifies the shape of the slicks. Therefore, in this case each oil spill source is not described by a single location, but by an ensemble of geographical points which identify a contour line.
The data required to define the initial oil spill condition are:

- location: i.e., the geographical coordinates of the oil source location. In the case of a detected slick, the polygon contour/s has to be specified;
- start time: i.e., the start time of the spill or the time of the slick detection;
- volume: i.e., the total amount of oil released in a spill or the total oil amount of the detected slick;
- duration: i.e., spill duration (for instantaneous or detected slicks, it is zero);
- age: i.e., the time period from initial arrival in the sea. This information can be provided by satellite monitoring systems.

An Euler forward scheme is utilized for numerical solution of the transport equation. The Lagrangian velocity field at the current time step at the particle position is computed performing a linear interpolation in time between successive input velocity fields, a bi-linear interpolation in horizontal space using the four external field grid points of the Eulerian model field (oceanographic or meteorological) nearest the particle position and a linear vertical interpolation between the two field at the Eulerian model levels nearest the particle depth.

4. Installing MEDSLIK-II v2.01

4.1 Software requirements

The list of software requirements is given below with a link to the source web page (the command lines refer to a Debian GNU/Linux Operating System).

1) The fortran compiler **gfortran** (https://gcc.gnu.org/fortran/)

```
sudo apt-get update
sudo apt-get install gcc gfortran
```

NOTE: gfortran may be pre-installed in your linux systems. Check the version using

```
-version
```

2) The **NetCDF libraries** (>v4.2) (https://www.unidata.ucar.edu/software/netcdf/)

The easiest way is to install the **precompiled** NetCDF library (NetCDF C and Fortran compiler)

```
udo apt-get install netcdf-bin
sudo apt-get install libnetcdf-dev
sudo apt-get install libnetcdff-dev
```

ompatibility problem with your Fortran compiler. The problem is that for a library to be compatible with your Fortran compiler it has to be compiled with the same compiler and with the same flags you will be using and that may be not the case when you get a precompiled library from a repository.
A safer way is to install NetCDF **from the source code**. The Fortran netCDF library need to be built after the NetCDF-C library is built and installed.

- Download the NetCDF source code from unidata website

```
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4.4.1.1.tar.gz
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-fortran-
4.4.4.tar.gz
```

- Download also the supporting libraries (HDF5, zlib, szip)

```
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4/hdf5-
1.8.13.tar.gz
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4/zlib-
1.2.8.tar.gz
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4/szip-2.1.tar.gz
```

- Start to install the szip library. Untar the file:

```
tar -zxvf szip-2.1.tar.gz
```

- Configure and build the szip library in the directory /usr/local/szip/szip-2.1

```
cd szip-2.1
./configure --prefix=/usr/local/szip/szip-2.1
make
make check
make install
```

- Then install the zlib library. Untar the file:

```
tar -zxvf zlib-1.2.8.tar.gz
```

- Configure and build the zlib library in the directory /usr/local/zlib/zlib-1.2.8

```
cd zlib-1.2.8
./configure --prefix=/usr/local/zlib/ zlib-1.2.8
make
make test
make install
```

- Then install the hdf5 library. Untar the file:

```
tar -zxvf hdf5-1.8.13.tar.gz
```

- Configure and build the hdf5 library in the directory /usr/local/hdf5/hdf5-1.8.13

```
cd hdf5-1.8.13
./configure --enable-fortran --with-zlib=/usr/local/zlib/zlib-1.2.8
--with-szip=/usr/local/szip/szip-2.1 --prefix=/usr/local/hdf5/hdf5-
1.8.13
make
make test
make install
```

- Now you can install the netcdf-library. First Netcdf-C library. Untar the file:

```
tar -zxvf netcdf-4.4.1.1.tar.gz
tar -zxvf netcdf-fortran-4.4.4.tar.gz
```

- Configure and build the NetCDF-C library in the directory /usr/local/netcdf/netcdf-4.4.1.1

```
cd netcdf-4.4.1.1
export LDFLAGS='-L/usr/local/zlib/zlib-1.2.8/lib -
L/usr/local/szip/szip-2.1/lib -L/usr/local/hdf5/hdf5-1.8.13/lib'
export CPPFLAGS='-I/usr/local/zlib/zlib-1.2.8/include -
I/usr/local/szip/szip-2.1/include -I/usr/local/hdf5/hdf5-
1.8.13/include'
./configure --prefix=/usr/local/netcdf/netcdf-4.4.1.1 --disable-dap
--disable-dap-remote-tests --enable-netcdf4 --enable-shared
make
make check
make install
```

- You need to add the path of the netcdf library to etc/ld.so.conf and update shared library cache. In this way, our library can be correctly found

```
echo "/usr/local/netcdf/netcdf-4.4.1.1/lib" >
/etc/ld.so.conf.d/netcdf-4.4.1.1.conf
sudo ldconfig
```

- Now you can install the netcdf-fortran library. Untar the file:

```
tar -zxvf netcdf-fortran-4.4.4.tar.gz
```

- Configure and install the NetCDF-fortran libraries

```
cd ../netcdf-fortran-4.4.4
export LDFLAGS='-L/usr/local/netcdf/netcdf-4.4.1.1/lib'
export CPPFLAGS='-I/usr/local/netcdf/netcdf-4.4.1.1/include'
./configure --prefix=/usr/local/netcdf/netcdf-4.4.1.1 --enable-
shared
make
make check
make install
```

- Finally add the /bin directory to the PATH environment variable. Insert the following command to the ~/.bashrc

```
export PATH=$PATH:/usr/local/netcdf/netcdf-4.4.1.1/bin
```

Check the version using

```
c-config --version
```

3) The Climate Data Operator (**CDO**) software (https://code.mpimet.mpg.de/projects/cdo)

```
sudo apt-get install cdo
```

4) The **Python 3.x**. The easiest and common way to install Python is to install Miniconda distribution (https://docs.conda.io/en/latest/miniconda.html)

```
wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-
x86_64.sh
chmod +x Miniconda3-latest-Linux-x86_64.sh
./Miniconda3-latest-Linux-x86_64.sh
```

Follow the prompts on the installer screens. Miniconda contains also the command-line tool called **conda** which is a package manager you can use to install additional python packages. Test your installation by run the command `conda list`. A list of installed packages appears if it has been installed correctly.

5) The additional python modules **netCDF4**, **matplotlib**, **scipy**, **basemap**, **basemap-data-hires**.

```
conda install -c conda-forge netcdf4
conda install -c conda-forge matplotlib
conda install -c conda-forge scipy
conda install -c conda-forge basemap
conda install -c conda-forge basemap-data-hires
```

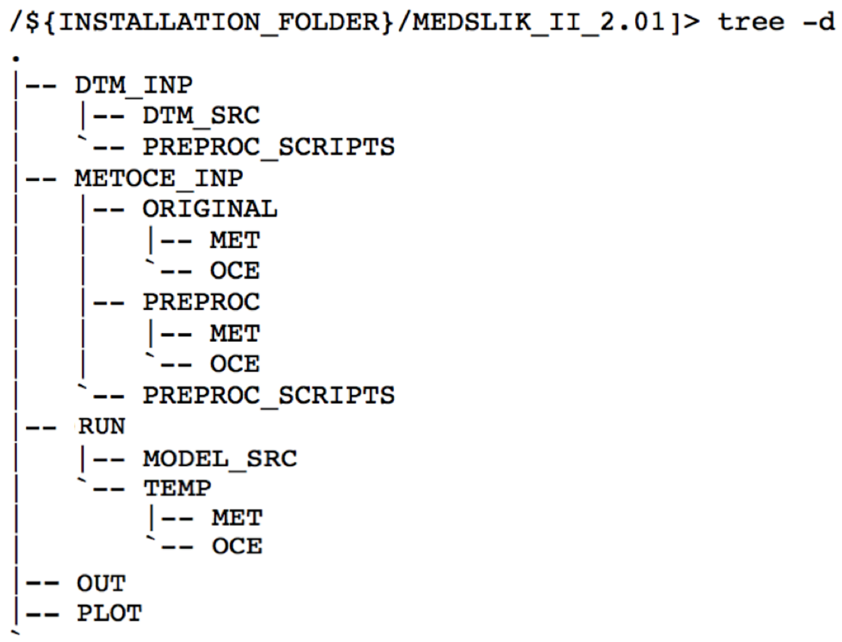
4.2 Downloading of the code

After registration four links, named Version_1.01, Version_1.02, Version_2.0, and Version_2.01 respectively, are accessible. Each link leads to the homepage of the corresponding version of the model: here there are the links to download the code, to download the user manual and to download all the data needed to run the test cases.
After downloading the version 2.01 code, the `zip` compressed archive has to be moved in user-selected installation path.

The variables `${DOWNLOAD_FOLDER}` and `${INSTALLATION_FOLDER`} need to be replaced by the user's system variables.
Because the installation process requires write, read and execution permissions, the installation directories need to have the necessary permits.

4.3 Structure of the model code directories

Extracting the `zip` file creates the `/${INSTALLATION_FOLDER}/MEDSLIK_II_2.01` directory tree which is organized with the following structure (Fig. 4):



```
/${INSTALLATION_FOLDER}/MEDSLIK_II_2.01]> tree -d
.
|-- DTM_INP
|    |-- DTM_SRC
|    `-- PREPROC_SCRIPTS
|-- METOCE_INP
|    |-- ORIGINAL
|    |    |-- MET
|    |    `-- OCE
|    |-- PREPROC
|    |    |-- MET
|    |    `-- OCE
|    `-- PREPROC_SCRIPTS
|-- RUN
|    |-- MODEL_SRC
|    `-- TEMP
|         |-- MET
|         `-- OCE
|-- OUT
|-- PLOT
`
```

Fig. 4 Model code directories structure.

The description of the meaning and function of each directory are given below:
`./DTM_INP/`: Digital Terrain Model (DTM) Input contains data and tools to generate the bathymetry and coastlines for the domain of user's interest.
`./DTM_INP/DTM_SRC/`: DTM Sources contain the global bathymetry and coastline data.
`./DTM_INP/PREPROC_SCRIPTS/`: Preprocessing Scripts contain the Python scripts for generating the user's domain bathymetry and coastlines in a proper ASCII format.
`./METOCE_INP/`: Meteo-Oceanographic Input contains data and tools to generate the meteo-oceanographic input files suitable for MEDSLIK-II v2.01.
`./METOCE_INP/ORIGINAL/`: Contains the original data downloaded from CMEMS (**`OCE`**) and ERA-Interim (**`MET`**).
`./METOCE_INP/PREPROC/`: Contains the preprocessed data on ocean currents and SST (**`OCE`**) and 10 m wind (**`MET`**) in the NetCDF format appropriate for `Extract_II.for`.
`./METOCE_INP/PREPROC_SCRIPTS/`: Preprocessing Scripts contain the Python scripts for generating the user's domain meteo-oceanographic data in the proper NetCDF formats.
`./RUN/`: Contains directories, files and running scripts to configure and run simulations.

`./RUN/MODEL_SRC/`: Model Sources contain the model source codes and compiler.
`./RUN/TEMP/`: Temporary meteo-oceanograpic input data contains directories with the meteo- (**`MET`**) and oceanographic (**`OCE`**) data in the ASCII format appropriate for `medslik_II.for`.
`./OUT/`: Contains the simulation output directories.
`./PLOT/`: Contains the Python scripts for visualization of the MEDSLIK-II v2.01 outputs.

4.4 Model source codes

The MEDSLIK-II v2.01 source codes are located in the directory of **`./RUN/MODEL_SRC/`**
`medslik_II.for` that simulates the oil spill transport and fate;
`Extract_II.for` that extracts the required input meteo-oceanographic data from netCDF files and transforms them into ASCII format appropriate for `medslik_II.for`.
The Fortran codes in the directory of `./RUN/MODEL_SRC/` also include:
`lat_lon.for` that calculates the geographical limits of the area affected by the spill (the area is calculated assuming the slick travels at less than 1.5 nautical miles/hr from the spill site. The computed geographical limits of the area affected by the spill are saved in the ./`RUN/medslik.tmp` file.
`jday.f` that converts the calendar dates to astronomical Julian dates and the elapsed time between instances over periods of time. More specifically, the script gets a day in the format `YYYYMMDD` as the first argument and returns a date in the same format adding or subtracting a number of days given as the second argument.
`ReadSaDatat_EMSA.py` reads the satellite data file (if the corresponding option has been selected) and writes two text files (`initial.txt` and `medslik_sat.inp`) containing the slick contour and oil spill data. This routine has to be adapted for the specific satellite date file format (usually XML or GML file).
Additionally, a Python script is inherited from MEDSLIK-II v1.01 as follows.
`read_oil_data.py` reads the oil characteristics from the oil-type database file `oilbase.txt` and `oil_list.txt` located in the `./RUN/` directory.

4.5 Compiling and linking the codes

Open
`/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/MODEL_SRC/compile.sh` and give the correct path of your `${INSTALLATION_FOLDER}` variable and netCDF installation.
Then, go to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/MODEL_SRC` and compile the code:
`sh compile.sh`.
The model should be compiled now. Two final steps:
go to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/medslik_II.sh` and put the correct MEDSLIK-II v2.01 installation folder to the `${INSTALLATION_FOLDER}` variable, repeat the same procedure for
`/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/RUN.sh`

5. Running MEDSLIK-II v2.01

5.1 Configuration file 1

`config1.txt` is a configuration file in text format. This file contains information about the initial slick or release point, the input currents, SST and winds and the oil type, etc. That file has to be declared by the user.
It has the same format as `medslik_inputfile.txt` in MEDSLIK-II v1.01 and MEDSLIK-II v2.0 versions, and contains the following bash variables:

`SIM_NAME`: the name of the simulation. Name of the simulation (any name without blank spaces and symbols), e.g., `SIM_NAME=TEST` or `SIM_NAME=paria_casestudy`. It will become a suffix of the name of the output directory.
`MODEL:` Specification of the oceanographic model that generates the ocean currents and SST, e.g., `MODEL=MRC` is the CMEMS `GLOBAL OCEAN 1/12° PHYSICS ANALYSIS AND FORECAST` dataset.
`WIND:` is the 10m wind which is now set as WIND=ERAi because the test case is set to run with ERA-Interim.
`sim_length`: The number of hours from the start of the spill or trajectory for which the prediction of position and state of the oil are required. This should be written using 4 characters, e.g., `sim_length=0072`.
`day`: The day on which the spill or the trajectory starts.
`month`: The month on which the spill or the trajectory starts. This variable must be written using 2 characters, e.g., `month=02`.
`year`: The year on which the spill or the trajectory starts. This variable must be written using 2 characters, e.g., `year=08`.
`hour`: The hour of day between 0 and 23. This variable must be written using 2 characters, e.g., `hour=04`.
`minutes`: The minutes after the hour of start, between 0 and 59, at which the spill or trajectory is started. This variable must be written using 2 characters, e.g., `minutes=05`.
`lat_degree`: The latitude at which the spill occurred is entered in degrees and decimal minutes. This variable has 2 characters, e.g., `lat_degree=42`.
`lat_minutes`: The latitude at which the spill occurred or the trajectory started is entered in degrees and decimal minutes. This variable has 4 characters, e.g., `lat_minutes=22.20`.
`lon_degree`: The longitude at which the spill occurred or the trajectory started is entered in degrees and decimal minutes. This variable has 2 characters, e.g., `lon_degree=10`.
`lon_minutes`: The longitude at which the spill occurred or the trajectory started is entered in degrees and decimal minutes. This variable has 4 characters, e.g., `lon_minutes=55.50`.
`duration`: The number of hours during which oil was spilling. For an instantaneous spill enter 0. It is written using 4 characters, e.g., `duration=0072` or `duration=0000`.
`spillrate`: The rate (in tons per hour) at which oil was spilt. For an instantaneous spill, the total volume has to be written, e.g., `spillrate=10.50`.

`age`: The slick age, which can be 0, 24 or 48 hrs, e.g., `age=0`.
`grid_size`: Each time the results are saved in one of the output files, the particles are aggregated or binned into uniform bines, the size of which is specified in m by the grid_size parameter, e.g., `grid_size=150`.
`OIL`: If the precise name of the oil is known, write *`OIL=NAME`*, if it is unknown write `OIL=API.` And specify API of the oil e.g.,
`OIL=API`
`OIL_TYPE=28`
The name and API of the oil must be specified in the file `oilbase.txt.`
In the same manner as MEDSLIK-II v1.01 MEDSLIK-II v2.0, and MEDSLIK-II v2.01, allows the simulation of transport and fate of any areal oil slick with manual insertion of contour coordinates or areal slick using satellite data.
`SAT_DATA=NO` and **`ContourSlick=NO`** specify the simulation of a point source of spill, when there is no need to fill in the number of slicks - `Nslick` and list of latitude and longitude of slick points contour - `S1lon[1]`, `S1lat[1]`, `S1lon[2]`, `S1lat[2]`...
`SAT_DATA=NO` and **`ContourSlick=YES`** should be applied to perform a simulation starting from the initial slick. If `ContourSlick=YES` is chosen, then the following variables have to be specified (otherwise leave them blank):
`Nslick`: Number of oil slick to be simulated, written using 1 character, e.g., `NSlick=2.` `S1lon[1], S1lat[1], S1lon[2], S1lat[2]...`). Fill in the list of latitude and longitude of slick points contour (latitude and longitude (in decimal degrees) of the oil slick polygon vertices.
`SAT_DATA=YES` and **`ContourSlick=NO`** specify the simulation of the areal source of spill using satellite data (no need to fill in `Nslick` and list of latitude and longitude of slick points contour - `S1lon[1], S1lat[1], S1lon[2], S1lat[2]...`).
`namefileGML` has to be filled in with the name of the GML file (e.g., `namefileGML=ASA_WSM_1PNACS20080806_095116_000000612071_00022_3 3643_0001.N1.00000_Oil.gml`). The file has to be saved in `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/`.
`N_OS` has to be filled with the number of the oil slick in the file to be simulated (e.g., N_OS=1).

5.2 Configuration file 2

`config2.txt` is a file, where the simulation parameters are set. Having the ASCII format, it can be read and edited using any text editor. The parameters can be changed if the user needs to try any values that are different from the default ones. It has the same format as `medslik5.par` in MEDSLIK-II v1.01 and MEDSLIK-II v2.0 versions, and comprehends the following bash variables:

`Stokes drift correction:` Choosing 01 allows the model to use the wave-induced velocity (the Stokes drift (Eq. 9)) calculated using an empirical formulation (De Dominicis et al., 2013a). Choosing 00 for the Stokes drift velocity will not be applied. The default value is 01.
`Drift Factor`: The drift speed of the slick is equal to this factor multiplied by the wind speed. The default value is 0.

`Drift Angle:` The wind-driven drift of the slick occurs at this angle to the right of the wind direction. The default value is 0.0 degrees, which causes the slick to move directly downwind.
`Variable Drift Angle:` Choosing 01 allows the model to use a drift angle that decreases as the wind speed increases; a wind speed at which the drift angle is reduced by 50% must then be entered in the line below. The default is that such a reduction is not made.
`Reduction of Forecast Wind Speed:` When using forecast water circulation in a simulation, the forecast water velocities (configuration file 1) already include the effect of the wind forces on the water surface. It may thus be considered appropriate in some cases to reduce the wind speed used in the drift formula by a fraction of the winds used in the forecast. This can be done writing 01 in effective wind speed and entering in the line below the reduction fraction (between 0 and 1). The default is that such a reduction is not made.
`Smagorinsky Scheme:` Choosing 01 allows the Lagrangian turbulent diffusion coefficient to be space dependent. The default value is 0.
`Horizontal Diffusivity:` The default value is 2.0 m^2/s. A larger value will cause the slick to spread wider.
`Vertical Diffusivities:` This is used in the transformation part, not in the advection: a larger value on top of the upper mixed layer and a smaller value below the mixed layer. The defaults are 0.01 m^2/s and 0.0001 m^2/s respectively.
`Depth of Mixed Layer:` The default for this depth is 30 m. This is used in the transformation part, not in the advection.
`Number of parcels:` 25 000 parcels are used as the default in the Monte Carlo algorithm to compute turbulent diffusion, but the user can increase this number up to 300 000.
`Depths of Currents:` In addition to the sea surface, the 3 depths (10 m, 30 m and 120 m) are available to calculate the dispersed fraction of oil in the water column.
`Selection of the depth currents for advection of slick:` It is possible to choose between 4 options: 00 for surface currents, 01 for 10 m deep currents, 02 for 30 m deep currents and 03 for 120 m deep currents. The default strategy is to use the surface velocities for advection of the slick and ignore the wind drift altogether (by setting the drift factor equal to zero).
`Number of Time Steps per Hour:` The default value is 2, which corresponds to a default time step of 30 minutes. In general, this is enough, but for a continuous spill with strong winds and/or currents, the 30-minute step may cause the computed slick to appear as a number of discrete sub-slicks that do not merge for several hours. This miscomputation can be solved by applying the shorter time step.
`Dimension of the Array used for representing the slick:` Each time the results are saved in one of the output files, the particles are aggregated inside 'bins', the size of which was entered in `medslik_inputfile.txt` as the variable `grid_size.` These bins form an array the dimension of which is set by this parameter. The default array dimension is 4000×4000.
The parameters from the sections describing the oil weathering: Evaporation Parameters, Emulsification Parameters, Dispersion Parameters, Spreading Parameters, Coastal Impact Parameters are discussed in details in De Dominicis et al. (2013a).

5.3 JSON configuration file for the Python scripts

You can invoke the Python scripts to run with a predefined configuration using the `conf.json` file to set parameters without modifying the code of each Python script. For example, you can preprocess currents with:
`$ python preproc_currents_mdk2.py conf.json`.

5.4 Model domain bathymetry and coastlines

Original global bathymetry and coastline datasets are provided in the directory
`./DTM_INP/DTM_SRC/` as follows
`GEBCO_2014_1D.nc` is a GEBCO database with an original 30” resolution; and
`gshhs_f.b` is taken from the GSHHG global coastline dataset
(https://www.ngdc.noaa.gov/mgg/shorelines/data/gshhg/latest/).
The Python scripts that generate the users’ domain bathymetry and coastlines are located in
`./DTM_INP/PREPROC_SCRIPTS/`.
Use **`preproc_gebco_mdk2.py`** to generate the bathymetry file with the extension of `.bath`.
Use the predefined configuration file `conf.json` to set parameters without modifying the code. It needs to be passed as an argument of the python scripts. For example
`$ python preproc_gebco_mdk2.py conf.json`.
Subversion: considering the wide range of areas and scales where oil spills can take place, it is very likely that GEBCO data will not fit all the scenarios or it may not be the most applicable solution. In those cases, the user is encouraged to use different data sources and it should not be that hard to build his/her own bathymetry file as long as the following guidelines are respected:
follow the file structure:

–line 1: a good-enough description of your bathymetry file. For instance, ”GEBCO 30sec - derived bathymetry for the Patagonian shelf”
–line 2: [`minlong`] [`maxlong`] [`minlat`] [`maxlat`] (space between variables with seven digits - 4 decimals - (in degrees))
–line 3: [`nx`] [`ny`] (number of columns (E-W) and rows (S-N) in your source 2D bathymetry field)
–following lines: [depth value] (positive for values below the water line) with no decimals (4 digits). In case of land, put 9999. The bathymetry vector is based on the 2D bathymetry fields looping like:

```
for i in n rows:
for j in n columns:
                    bat1d.append(bat2d[i,j])
```

For further understanding, check the procedure done in the Python script available at our website.

Use `preproc_gshhs_mdk2.py` to generate the coastline file with the extension of `.map` and `.txt`.
Use the predefined configuration file `conf.json` to set parameters without modifying the code. It needs to be passed as an argument of the python scripts. For example, `$ python preproc_gshhs_mdk2.py conf.json`.

Depending on the spatial scale of your study and on the number of simulations you are expected to perform, you may consider whether to use the full resolution file (more points and improved representation of the coastline and longer simulations) or high/intermediate (less points, poorer representation of the coastline but faster simulations) resolution.
Subversion: Very high-resolution applications are becoming more and more common in oceanography and, just like in the bathymetry case, the user may be interest in using his/her own coastline map in MEDSLIK-II v2.01. Again, it should not be that hard to build his/her own coastline file as long as the following guidelines are respected:
be sure your polygons never remain open.
be sure it follows the file structure:

- line 1: [number of coastline polygons, `p`, in your study area - 4 digits]
- line 2: [number of points, `n`, forming the polygon `p1`] [0]
- line 3: [`p1x`] [`p1x`] - 8 digits with 5 decimals
- line 4: [`p2x`] [`p2x`] - 8 digits with 5 decimals
- line 3+n: [`pnx`] [`pnx`] - 8 digits with 5 decimals
- line 3+n+1: [number of points, `n`, forming the polygon `p2`] [`0`]
- repeat procedure writing polygon points
- this sequence should be repeated for all the polygons in your study area

For further understanding, check the procedure done in the Python script available in our website.

5.5 Model meteo-oceanographic inputs

Similar to MEDSLIK-II v2.0, MEDSLIK-II v2.01 supports one single type of ocean field format and one type of atmospheric field format. The user will be responsible for translating his/her input data into MEDSLIK-II v2.01 format and he/she is strongly encouraged to perform this before using the Fortran code `Extract_II.for`.
Data and tools to generate the meteo-oceanographic input files suitable for MEDSLIK-II v2.01 are located in **`./METOCE_INP/`**.

First put GLOBAL OCEAN 1/12° PHYSICS ANALYSIS AND FORECAST data downloaded from http://marine.copernicus.eu/services-portfolio/access-to-products/?option=com_csw&view=details&product_id=GLOBAL_ANALYSIS_FORECAST_PHY_001_024 in **`./METOCE_INP/ORIGINAL/OCE/`**. Then, put the atmospheric ERA-Interim data downloaded from https://apps.ecmwf.int/datasets/data/interim-full-daily/levtype=sfc/ in **`./METOCE_INP/ORIGINAL/MET/`**.
The Python script **`preproc_currents_mdk2.py`** preprocesses the original ocean currents' and SST data, pre-cropping them in space and time and converting the NetCDF format to the appropriate one to be read from `Extract_II.for`.
Use the predefined configuration file `conf.json` to set parameters without modifying the code. It needs to be passed as an argument of the python scripts. For example, `$ python preproc_currents_mdk2.py conf.json`.
Check the output in the directory of **`./METOCE_INP/PREPROC/OCE`**.

The Python script **`preproc_winds_mdk2.py`** preprocesses the original 10 m wind data, pre-cropping them in space and time and converting to the NetCDF format appropriate for `Extract_II.for`.

Use the predefined configuration file `conf.json` to set parameters without modifying the code. It needs to be passed as an argument of the python scripts. For example, `$ python preproc_winds_mdk2.py conf.json`.
Check the output in the directory of **`./METOCE_INP/PREPROC/MET`**.

In case the user wishes to employ different data sources, he/she will have to translate them into the appropriate format. Oceanographic fields are divided into three different files (meridional current, zonal current and sea surface temperature). The model requires currents at four different depths: 0m, 10m, 30m and 120m.
Temperature:

- File name: **`MDK_ocean_yymmdd_T.nc`**, where `yy` correspond to the last two digits of the year, mm month and `dd` day. Daily files include hourly fields.
- Dimensions: `time_counter` for time, `y` for latitude and `x` for longitude. Latitude starting from the southernmost grid point. Longitude starting from the westernmost point.
- Variables: `nav_lat` and `nav_lon` for latitude and longitude fields dependent on the `y` and `x` dimensions, respectively. The temperature fields are included in `votemper` which depends on (`time_counter, y, x`).
- Temporal resolution: hourly starting from 13:00.
- Missing value: 9999
- Units: degrees Celsius

Zonal currents:

- File name: **`MDK_ocean_yymmdd_U.nc`**, where `yy` correspond to the last two digits of the year, mm month and `dd` day. Files are daily containing hourly fields.
- Dimensions: `time_counter` for time, `y` for latitude and `x` for longitude, and `depthu` for depth. Latitude starting from the southernmost grid point. Longitude starting from the westernmost point.
- Variables: `nav_lat` and `nav_lon` for latitude and longitude fields dependent on (`y`, `x`) dimensions. The current fields are included in vozocrtx which depends on (time_counter, depthu, y, x).
- Temporal resolution: hourly starting from 13:00.
- Missing value: 9999
- Units: m/s

Meridional currents:

- File name: MDK_ocean_yymmdd_V.nc, where yy correspond to the last two digits of the year, mm month and dd day. Files are daily containing hourly fields.

- Dimensions: time_counter for time, y for latitude and x for longitude, and depthv for depth. Latitude starting from the southernmost grid point. Longitude starting from the westernmost point.
- Variables: nav_lat and nav_lon for latitude and longitude fields dependent on (y, x) dimensions. The current fields are included in vomecrty which depends on (time_counter, depthv, y, x).
- Temporal resolution: hourly starting from 13:00.
- Missing value: 9999
- Units: m/s

Wind fields are given using a single file containing meridional and zonal 10m winds in m/s. Currently, the model reads hourly resolution wind fields. The wind files are organized as follows:

- File name: YYYYmmdd.nc, where YYYY correspond to all the four digits of the year, mm month and dd day. Files are daily.
- Dimensions: time for time, lat for latitude and lon for longitude. Latitude starting from northernmost grid point. Longitude starting from westernmost point.
- Variables: lat and lon for latitude and longitude fields dependent on (lat) and (lon) dimensions, respectively. The wind fields are included in U10M and V10M and depend on (time, lat, lon).
- Temporal resolution: hourly starting from 00:00.
- Missing value: 9999
- Units: m/s

5.6 Running the simulations

Be sure that the dimensions of your meteo-oceanographic input data are written in a proper way in `Extract_II.for`. Adjust the Strings #83 and #499, and re-compile `Extract_II.for`, if that is needed. Now, you are ready to run the code and you should use: `./RUN.sh`.

5.7 Model output and visualization scripts

The output directories are located in **`./OUT/`**. The MEDSLIK-II v2.01 output format is inherited from MEDSLIK-II v2.0. The output file **`spill_properties.nc`** contains information relative to the evolution of each oil parcel at each time step. The output file is built as follows:

- File name: `spill_properties.nc`
- Dimensions: `parcel_id` refers to the parcel number and `time` refers to simulation time steps.

Variables:

– `latitude`: latitude of each oil parcel (degrees)

– `longitude`: longitude of each oil parcel (degrees)

– `evaporative_volume`: parcel volume relative to the evaporative component of the oil (m^3)

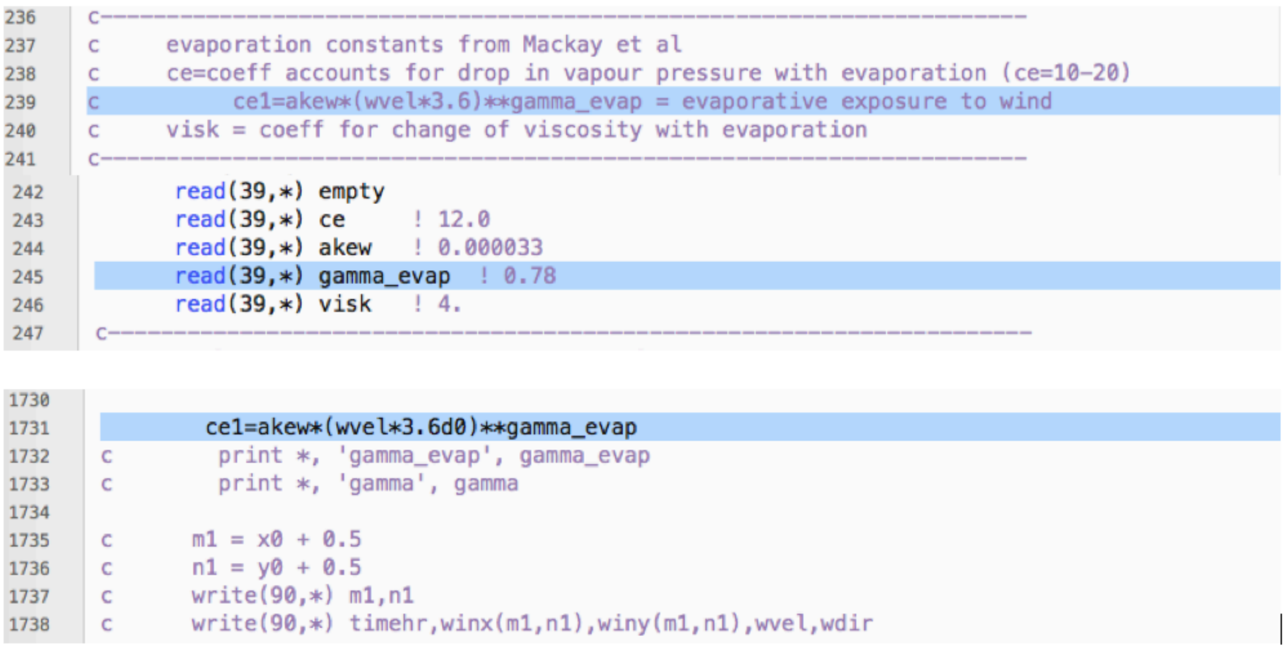
– `non_evaporative_volume`: parcel volume relative to the non-evaporative component of the oil (m^3)
– `water_fraction`: percentage of water in each parcel (%)
– `particle_status`:
    0: not released
    1: released and spreading
    2: released and not spreading
    3: released and dispersed
    4: bottom
    -n = parcel is beached. ”n” corresponds to the total volume of oil seeped at the coastline.
    9: beyond boundary limits
– `time`: time index
– `segment_id`: segment index
– `total_fixed_oil`: total amount of oil beached and fixed at the coast (tonnes)
– `viscosity_emulsion_1`: viscosity emulsion water-oil for mini-spill 1 (P a.s)
– `viscosity_emulsion_2`: viscosity emulsion water-oil for mini-spill 2 (P a.s)
– `viscosity_oil_1`: viscosity oil for mini-spill 1 (P a.s)
– `viscosity_oil_2`: viscosity oil for mini-spill 2 (P a.s)
– `density_emulsion_1`: density water-oil for mini-spill 1 (kg/m^3)
– `density_emulsion_2`: density water-oil for mini-spill 2 (kg/m^3)
– `water_fraction_1`: percentage of water in the emulsion for mini-spill 1 (%)
– `water_fraction_2`: percentage of water in the emulsion for mini-spill 2 (%)
– `volume_ratio`: volume ratio water/oil (%)

Two Python scripts are provided in the **`./PLOT/`** directory to plot the surface (**`oil_track_mdk2.py`**) and beached (**`oil_beached_mdk2.py`**) oil at each time step. Use the predefined configuration file `conf.json` to set parameters without modifying the codes. For example, `$ python oil_track_mdk2.py conf.json`.

6. Differences with v2.00

6.1 Bug fixes

CNR-IAS team found a problem when the different variables were defined by the same name `gamma`. This error lead to the incorrect calculation of the evaporation term in Eq. 2 and subsequent overestimation its contribution to the oil mass balance. It was found that the error mostly effected the light oil. To solve the problem, instead of the `gamma` in the evaporative part, the new variable `gamma_evap` was introduced. The corrected strings in `medslik_II.for` look as shown in Fig. 5:



```
c-----------------------------------------------------------------------
c     evaporation constants from Mackay et al
c     ce=coeff accounts for drop in vapour pressure with evaporation (ce=10-20)
c         ce1=akew*(wvel*3.6)**gamma_evap = evaporative exposure to wind
c     visk = coeff for change of viscosity with evaporation
c-----------------------------------------------------------------------
      read(39,*) empty
      read(39,*) ce      ! 12.0
      read(39,*) akew    ! 0.000033
      read(39,*) gamma_evap   ! 0.78
      read(39,*) visk    ! 4.
c-----------------------------------------------------------------------
```

```

            ce1=akew*(wvel*3.6d0)**gamma_evap
c            print *, 'gamma_evap', gamma_evap
c            print *, 'gamma', gamma

c         m1 = x0 + 0.5
c         n1 = y0 + 0.5
c         write(90,*) m1,n1
c         write(90,*) timehr,winx(m1,n1),winy(m1,n1),wvel,wdir
```

Fig. 5 Bug correction in `medslik_II.for`.

6.2 Upgrade of Python scrips

The Python scripts are ported from Python 2 to Python 3. The configuration file `conf.json` is implemented for the scripts:

`preproc_gebco_mdk2.p`
`preproc_gshhs_mdk2.py`
`preproc_currents_mdk2.py`
`preproc_winds_mdk2.py`
`oil_track_mdk2.py`
`oil_beached_mdk2.py`.

7. The *Paria* Test Case

7.1 Downloading the test case

In order to run the test case, download and decompress the test case file `paria_casestudy.zip` (11Gb). Copy the configuration file `conf.json` in the MEDSLIK-II 2.01 root directory (i.e., `/${INSTALLATION_FOLDER}/MEDSLIK_II_2.01/`).

7.2 Meteo-oceanographic inputs for the test case

Download the oceanographic model output from http://marine.copernicus.eu/services-portfolio/access-to-products/?option=com_csw&view=details&product_id=GLOBAL_ANALYSIS_FORECAST_PHY_001_024
using the instructions on http://marine.copernicus.eu/faq/what-are-the-advantages-of-each-cmems-download-mechanism/ .
Check them with the reference files located in
`paria_casestudy/METOCE_INP/ORIGINAL/OCE/`.
Preprocess the original data using `preproc_currents_mdk2` and check your results with the reference files located in `paria_casestudy/METOCE_INP/PREPROC/OCE/`. Before executing the script, open this file with any editor and give the correct path to the `INSTALLATION_FOLDER` variable. Use the predefined configuration file `conf.json` to set parameters without modifying the python scripts. For example,

```
$ python preproc_currents_mdk2.py conf.json
```

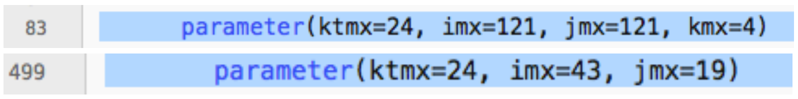
Before running the test case, be sure that these pre-processed files are copied to
`/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/METOCE_INP/PREPROC/OCE`.
Download the atmospheric model output from: https://apps.ecmwf.int/datasets/data/interim-full-daily/levtype=sfc/. Compare with the reference files located in
`paria_casestudy/METOCE_INP/ORIGINAL/MET/`. These files were downloaded for April 2017; selected time of 00:00:00; 06:00:00; 12:00:00; 18:00:00; selected step of 0; and selected parameters of 10 metre *U* wind component and 10 metre *V* wind component. For further information about ERA-Interim datasets see Berrisford et al. (2011) (https://www.ecmwf.int/en/elibrary/8174-era-interim-archive-version-20).
Preprocess the original data using `preproc_winds_mdk2` and check your results with the reference files located in `paria_casestudy/METOCE_INP/PREPROC/MET/`. Before executing the script, open this file with any editor and give the correct path to the `INSTALLATION_FOLDER` variable. Use the predefined configuration file `conf.json` to set parameters without modifying the code. Before running the test case, be sure that these pre-processed files are copied to
`/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/METOCE_INP/PREPROC/MET`.

7.3 Test case bathymetry and coastlines

Generate the bathymetry of the Venezuela shelf area using **`preproc_gebco_mdk2.py`**. Before executing the script, open this file with any editor and give the correct path to the `INSTALLATION_FOLDER` variable. Compare your result with a reference file `vnzl_.bath` found in `paria_casestudy/DTM_INP/`. Generate the coastline of the Venezuela shelf area using **`preproc_gshhs_mdk2.py`**. Before executing the script, open this file with any editor and give the correct path to the `INSTALLATION_FOLDER` variable. Use the predefined configuration file `conf.json` to set parameters without modifying the code. Compare your results with the reference files `vnzl.map` and `vnzl.txt` found in `paria_casestudy/DTM_INP/`. Before running the test case, be sure that these files are copied to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/DTM_INP/`.

7.4 Running the test case

Be sure that the dimensions of your meteo-oceanographic input data are written in a proper way in `Extract_II.for`. Adjust the Strings #83 and #499 as follows:



```
83        parameter(ktmx=24, imx=121, jmx=121, kmx=4)
499         parameter(ktmx=24, imx=43, jmx=19)
```

and re-compile `Extract_II.for`, if that is needed. Be sure that all the Fortran model source codes located in `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/ MODEL_SRC/` are compiled. The 120h-long simulation represents the so called *Paria* test case happened off the Venezuelan coast (see Table 2 for further information on the spill simulation scenario). Copy the initial condition (`config1.txt`) and model parameter (`config2.txt`) files found in `paria_casestudy/RUN` to the directory of `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/`.
Go to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/` and run MEDSLIK-II v2.01 with `./RUN.sh.` Check the medslik log file to check possible error/warning.

Table 2 MEDSLIK-II v2.01 simulation scenario applied to the *Paria* test case.

Spill time	24/04/2017 13:00
Spill position	10.81N -63.55W
Spill duration	120 h
Spill volume	180 tons
Oil API	28
Simulation length	48 h
Ocean fields	CMEMS GLOBAL_ANALYSIS_FORECAST_PHY_001_024
Wind fields	ERA-Interim
Stokes drift	MEDSLIK-II JONSWAP formulation
Number of parcels	25,000

7.5 Test case output and its visualization

Go to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/OUT/` and compare your result with the reference directory located in `paria_casestudy/OUT/`. Visualize your results using `oil_track_mdk2` and `oil_beached_mdk2.py` and compare them with the reference plots located in `paria_casestudy/OUT/ MERCATOR_global_2017_04_24_1300_paria_casestudy/plots`. Before executing the scripts, open this file with any editor and give the correct path to the INSTALLATION_FOLDER variable. Use the predefined configuration file `conf.json` to set parameters without modifying the codes.

Your results should look similar to Fig. 6.

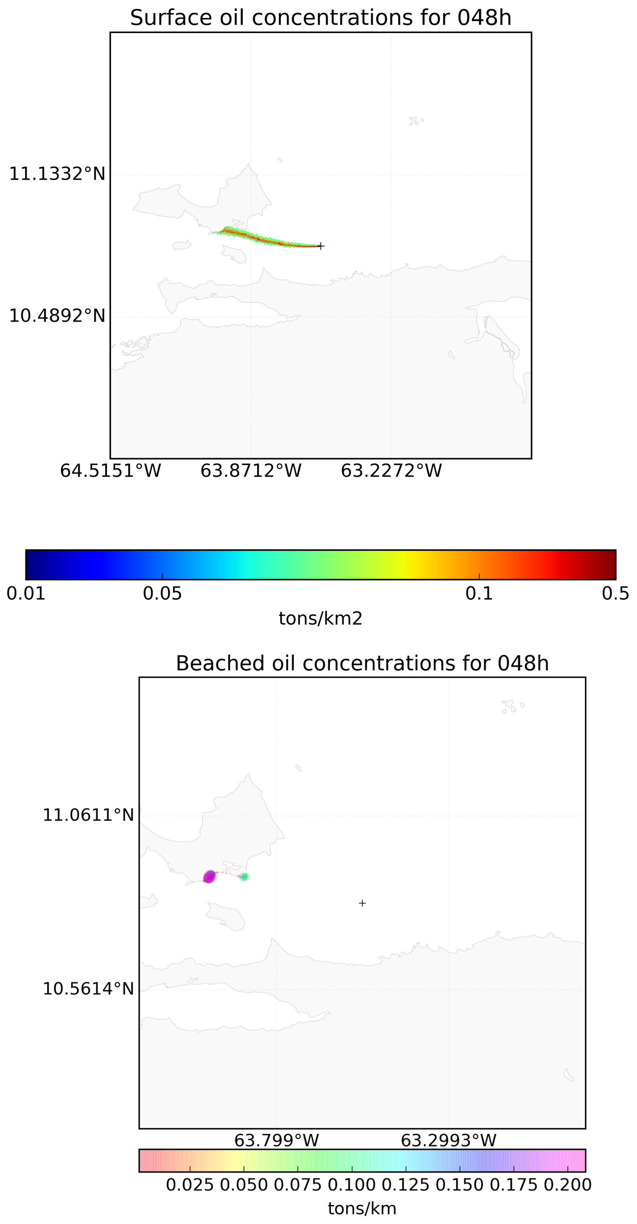


Fig. 6 Surface (tons/km^2) and beached (tons/km) oil concentrations after 48h of simulation for the *Paria* test case. Black cross marks the origin of the spill.

8. The Lebanon Test Case

8.1 Downloading the test case

In order to run the test case, download and decompress the test case file `lebanon_casestudy.zip` (30Gb). Copy the configuration file `conf.json` in the MEDSLIK-II 2.01 root directory (i.e., `/${INSTALLATION_FOLDER}/MEDSLIK_II_2.01/`).

8.2 Meteo-oceanographic inputs for the test case

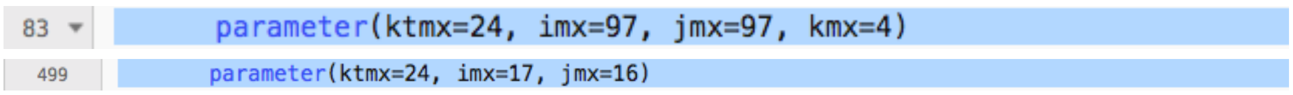
Download the oceanographic model output from http://marine.copernicus.eu/services-portfolio/access-to-products/?option=com_csw&view=details&product_id=
GLOBAL_REANALYSIS_PHY_001_030
using the instructions on http://marine.copernicus.eu/faq/what-are-the-advantages-of-each-cmems-download-mechanism/ .
Check them with the reference files located in
`lebanon_casestudy/METOCE_INP/ORIGINAL/OCE/`.
Preprocess the original data and check your results with the reference files located in
`lebanon_casestudy /METOCE_INP/PREPROC/OCE/`.
Before running the test case, be sure that these pre-processed files are copied to
`/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/METOCE_INP/PREPROC/OCE`.
Download the atmospheric model output from: https://apps.ecmwf.int/datasets/data/interim-full-daily/levtype=sfc/. Compare with the reference files located in
`lebanon_casestudy/METOCE_INP/ORIGINAL/MET/`. These files were downloaded for July and August 2006; selected time of 00:00:00; 06:00:00; 12:00:00; 18:00:00; selected step of 0; and selected parameters of 10 metre *U* wind component and 10 metre *V* wind component. For further information about ERA-Interim datasets see Berrisford et al. (2011)
(https://www.ecmwf.int/en/elibrary/8174-era-interim-archive-version-20).
Preprocess the original data and check your results with the reference files located in
`lebanon_casestudy/METOCE_INP/PREPROC/MET/`.
Before running the test case, be sure that these pre-processed files are copied to
`/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/METOCE_INP/PREPROC/MET`.

8.3 Test case bathymetry and coastlines

Generate the bathymetry of the Lebanon coastal area using **`preproc_gebco_mdk2.py`**. Compare your result with a reference file `lebn_.bath` found in `lebanon_casestudy/DTM_INP/`.
Generate the coastline of the Venezuela shelf area using **`preproc_gshhs_mdk2.py`**. Compare your results with the reference files `lebn.map` and `lebn.txt` found in
`lebanon_casestudy/DTM_INP/`. Before running the test case, be sure that these files are copied to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/DTM_INP/`. Use the predefined configuration file `conf.json` to set parameters without modifying the code. It needs to be passed as an argument of the python scripts.

8.4 Running the test case

Be sure that the dimensions of your meteo-oceanographic input data are written in a proper way in `Extract_II.for`. Adjust the Strings #83 and #499 as follows:



```
83        parameter(ktmx=24, imx=97, jmx=97, kmx=4)
499       parameter(ktmx=24, imx=17, jmx=16)
```

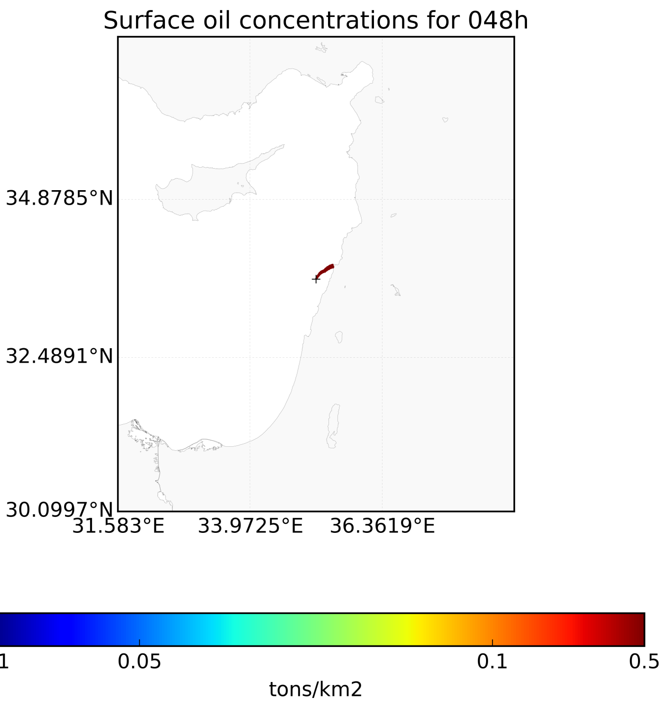
and re-compile `Extract_II.for`, if that is needed. Be sure that all the Fortran model source codes located in `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/ MODEL_SRC/` are compiled. The 480h-long simulation represents the so-called Lebanon test case happened off the Lebanon coast (see Table 2 for further information on the spill simulation scenario). Copy the initial condition (`config1.txt`) and model parameter (`config2.txt`) files found in `lebanon_casestudy/RUN` to the directory of `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/`.
Go to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/RUN/` and run MEDSLIK-II v2.01 with `./RUN.sh.`
Check the medslik log file to check possible error/warning.

Table 3 MEDSLIK-II v2.01 simulation scenario applied to the Lebanon test case.

Spill time	13/07/2006 08:00
Spill position	33.68N 35.17W
Spill duration	144 h
Spill volume	18770.4 tons
Oil API	20
Simulation length	480 h
Ocean fields	GLOBAL_REANALYSIS_PHY_001_030
Wind fields	ERA-Interim
Stokes drift	MEDSLIK-II JONSWAP formulation
Number of parcels	25,000

8.5 Test case output and its visualization

Go to `/${INSTALLATION_FOLDER}/MEDSLIK_II_v2.01/OUT/` and compare your result with the reference directory located in `lebanon_casestudy/OUT/`. Visualize your results and compare them with the reference plots located in `lebanon_casestudy/OUT/ MERCATOR_global_2006_07_13_0800_lebanon_casestudy/plots`. your results should look similar to Fig. 7.



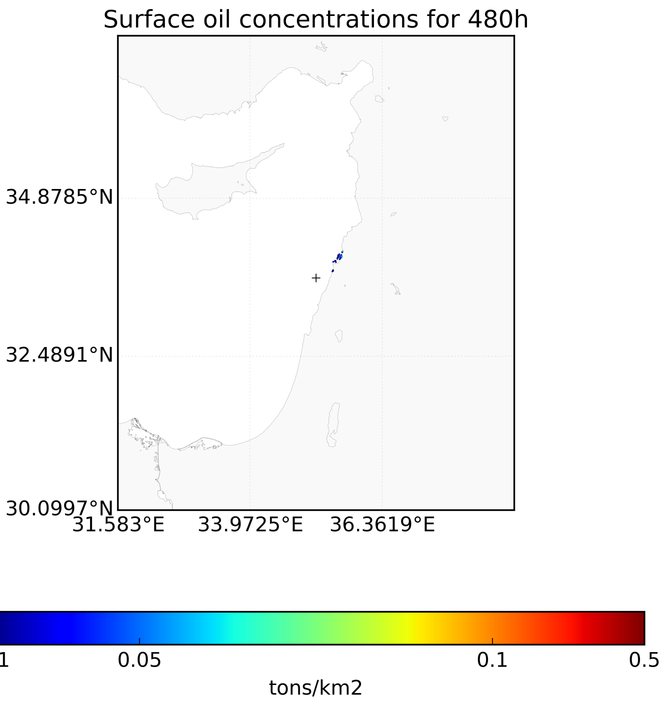


Fig. 6 Surface (tons/km^2) oil concentrations after 48h and 480h of simulation for the Lebanon test case. Black cross marks the origin of the spill.

References

BERRISFORD P., DEE D., POLI, P., BRUGGE, R., FIELDING, M., FUENTES, M., KÅLLBERG, P.W., KOBAYASHI, S., UPPALA, S., SIMMONS, A. 2011. THE ERA- INTERIM ARCHIVE VERSION 2.0. ECMWF, ERA REPORT SERIES, PP 23.

BRUCIAFERRI ET AL., 2015. MEDSLIK-II V1.02 USER MANUAL. HTTP://MEDSLIK-II.ORG/DATA/MODEL/MEDSLIK_II_V1.02_USER_MANUAL.PDF

DE DOMINICIS, M., 2012. MEDSLIK-II V1.01 USER MANUAL. HTTP://MEDSLIK-II.ORG/USERS/CODE/MEDSLIKII_1.01_USERMANUAL.PDF

DE DOMINICIS, M., PINARDI, N., ZODIATIS, G., AND LARDNER, R. 2013A, MEDSLIK-II, A LAGRANGIAN MARINE SURFACE OIL SPILL MODEL FOR SHORT-TERM FORECASTING – PART 1: THEORY, GEOSCI. MODEL DEV., 6, 1851–1869, DOI:10.5194/GMD-6-1851-2013.

DE DOMINICIS, M., PINARDI, N., ZODIATIS, G., AND ARCHETTI, R., 2013B, MEDSLIK-II, A LAGRANGIAN MARINE SURFACE OIL SPILL MODEL FOR SHORT-TERM FORECASTING – PART 2: NUMERICAL SIMULATIONS AND VALIDATIONS, GEOSCI. MODEL DEV., 6, 1871–1888, DOI: 10.5194/GMD-6-1871-2013

SEPP-NEVES, A.A., PINARDI, N., MARTINS, F., 2016. IT-OSRA: APPLYING ENSEMBLE SIMULATIONS TO ESTIMATE THE OIL SPILL RISK ASSOCIATED TO OPERATIONAL AND ACCIDENTAL OIL SPILLS, OCEAN DYNAMICS 66, 939–954.

SEPP-NEVES, A.A., 2018. MEDSLIK-II V2.0 USER MANUAL. HTTP://MEDSLIK-II.ORG/DATA/V2.0/MEDSLIK_II_V2.0_USER_MANUAL.PDF