

MAECS

Generated by Doxygen 1.8.4

March 12, 2014

Contents

Symbols	2
1 MAECS: Model for Adaptive Ecosystems in Coastal Seas	4
1.1 General Overview	4
2 Todo List	5
3 Data Type Index	6
3.1 Class Hierarchy	6
4 Data Type Index	7
4.1 Data Types List	7
5 File Index	8
5.1 File List	8
6 Data Type Documentation	9
6.1 fabm_hzg_maecs Module Reference	9
6.1.1 Detailed Description	9
6.1.2 Member Function/Subroutine Documentation	10
6.1.2.1 initialize	10
6.1.2.2 get_light_extinction	11
6.1.2.3 maecs_init_stoichvars	11
6.1.2.4 maecs_do	11
6.1.2.5 maecs_get_vertical_movement	13
6.2 maecs_functions Module Reference	13
6.2.1 Detailed Description	14
6.2.2 Member Function/Subroutine Documentation	14
6.2.2.1 calc_internal_states	14
6.2.2.2 calc_sensitivities	15
6.2.2.3 uptflex	15
6.2.2.4 foptupt	15
6.2.2.5 queuefunc	16
6.2.2.6 queuefunc1	16
6.2.2.7 queuederiv	16
6.2.2.8 sinking	16
6.2.2.9 min_mass	16
6.2.2.10 smooth_small	17
6.3 maecs_grazing Module Reference	17
6.3.1 Detailed Description	17
6.3.2 Member Function/Subroutine Documentation	18
6.3.2.1 grazing	18
6.3.2.2 grazing_losses	18
6.4 maecs_primprod Module Reference	18
6.4.1 Detailed Description	18
6.4.2 Member Function/Subroutine Documentation	18
6.4.2.1 photosynthesis	18
6.5 maecs_types Module Reference	19

6.5.1	Detailed Description	20
6.6	maecs_types::stoich_pointer Type Reference	20
6.6.1	Detailed Description	20
6.7	fabm_hzg_maecs::type_hzg_maecs Type Reference	20
6.7.1	Detailed Description	20
6.8	maecs_types::type_maecs_allocation_fractions Type Reference	21
6.8.1	Detailed Description	21
6.9	maecs_types::type_maecs_base_model Type Reference	21
6.9.1	Detailed Description	21
6.10	maecs_types::type_maecs_basic_traits Type Reference	21
6.10.1	Detailed Description	21
6.11	maecs_types::type_maecs_env Type Reference	21
6.11.1	Detailed Description	21
6.12	maecs_types::type_maecs_life Type Reference	22
6.12.1	Detailed Description	22
6.13	maecs_types::type_maecs_nutindex Type Reference	22
6.13.1	Detailed Description	22
6.14	maecs_types::type_maecs_om Type Reference	22
6.14.1	Detailed Description	22
6.15	maecs_types::type_maecs_phy Type Reference	23
6.15.1	Detailed Description	23
6.16	maecs_types::type_maecs_rhs Type Reference	23
6.16.1	Detailed Description	23
6.17	maecs_types::type_maecs_sensitivities Type Reference	23
6.17.1	Detailed Description	23
6.18	maecs_types::type_maecs_switch Type Reference	23
6.18.1	Detailed Description	23
6.19	maecs_types::type_maecs_traitdyn Type Reference	24
6.19.1	Detailed Description	24
6.20	maecs_types::type_maecs_zoo Type Reference	24
6.20.1	Detailed Description	24
7	File Documentation	25
7.1	maecs_functions.F90 File Reference	25
7.1.1	Detailed Description	25
7.2	maecs_grazing.F90 File Reference	25
7.2.1	Detailed Description	25
7.3	maecs_maecsd_combined.F90 File Reference	26
7.3.1	Detailed Description	26
7.4	maecs_primprod.F90 File Reference	26
7.4.1	Detailed Description	26
7.5	maecs_types.F90 File Reference	26
7.5.1	Detailed Description	27
8	Textual Model Description	28
8.1	Model structure	28
8.1.1	Mass equations	28
8.2	Process Descriptions	29
8.2.1	Growth rate components and primary production	29
8.2.2	Respiration and temperature	30
8.2.3	Multi-nutrient co-limitation	30
8.2.4	Uptake system allocation and PI-coefficients	31
8.2.5	Nutrient uptake	32
8.2.6	Grazing	33
8.2.7	Sinking loss	33
8.2.8	Particle aggregation	34
8.2.9	Other stuff	34
8.3	Adaptive trait regulation and differential trade-off	34
8.3.1	Growth derivatives under co-limitation	35

CONTENTS	1
8.3.2 Quota-uptake feed-back	36
8.3.3 Uptake activity regulation	37
8.3.4 Costs in P and Si uptake	37
8.3.5 Photoacclimation and transport	38
8.4 Acknowledgements	38
Bibliography	38
Index	40

Symbols

Symbols used in formulas, respective model parameters, and their descriptions

$1/h$	syn_nut synchrony n_queue in nutrient quota limitation
α	alpha specific light adsorption by chloroplasts
δ_θ	adap_theta adap_theta
δ_R	adap_rub adap_rub
Δt_v	tau_regV tau_regV
a_{chl}	a_chl attenuation coefficient due to Chl absorption
a_{spm}	a_spm attenuation coefficient of SPM
a_{water}	a_water background attenuation coefficient
Ae_{all}	Ae_all Activation energy
basal _{resp_{zoo}}	basal_resp_zoo basal respiration
const _{N_{C_{zoo}}}	const_NC_zoo zooplankton N:C ratio
const _{P_{C_{zoo}}}	const_PC_zoo zooplankton P:C ratio
detC _i initial	detC_initial Detritus Carbon
detN _i initial	detN_initial Detritus Nitrogen
detP _i initial	detP_initial Detritus Phosphorus
detS _i initial	detS_initial Detritus Silicon
dil	dil dilution of all concentrations except dissolved inorganics
DIN	nutN_initial Dissolved Inorganic Nitrogen DIN
domC _i initial	domC_initial Dissolved Organic Carbon
domN _i initial	domN_initial Dissolved Organic Nitrogen
domP _i initial	domP_initial Dissolved Organic Phosphorus
exud _{p_{hy}}	exud_phy phytoplankton exudation per production
frac _{P_{AR}}	frac_PAR photosynthetically active fraction of light
g_{max}	g_max maximum grazing rate
hydrol	hydrol hydrolysis rate
k_{grazC}	k_grazC half saturation graing
mort _{zoo}	mort_zoo quadratic mortality
NutOrder	NutOrder element order of recursive scheme. lower digit: synchrony element
phi _{agg}	phi_agg quadratic aggregation rate
phyN _i initial	phyN_initial Phytoplankton Nitrogen
phyP _i initial	phyP_initial Phytoplankton Phosphorus
phyS _i initial	phyS_initial Phytoplankton Silicon
Phy _C	phyC_initial Phytoplankton Carbon
Phy _P	nutP_initial Dissolved Inorganic Phosphorus DIP
Phy _{Si}	nutS_initial Dissolved Inorganic Silicon Si

$rel_{chloropl_min}$	$rel_chloropl_min$ chloroplast-C to phy-C ratio
$remin$	$remin$ pelagic remineralisation
$small$	$small$ lower limit for denominator in ratios; $small_finite=sqrt(small)$
T_{ref}	T_ref reference temperature
vS_{det}	vS_det sinking velocity for detritus
vS_{phy}	vS_phy sinking velocity for phytoplankton
$yield_{zoo}$	$yield_zoo$ yield of herbivory
$zooC_{initial}$	$zooC_initial$ Zooplankton Carbon
$zstoich_{pN}$	$zstoich_PN$ P-stoichiometry of active compounds(-> P costs)
σ	σ sigma Q-dependency of Rubisco activity/chloroplast ratio
θ_C	θ_{LHC} Chlorophylla-to-C ratio of LHC
ζ_{CN}	ζ_{CN} respiratory costs of N-synthesis/NO3-reduction
A_N^0	$AffN$ N-Affinity
A_P^0	$AffP$ P-Affinity
A_{Si}^0	$AffSi$ Si-Affinity
$f_{\theta} \theta_C$	$frac_chl_ini$ Chl:C ratio
f_R	$frac_Rub_ini$ fraction of Rubisco
P_{max}	P_max maximum potential photosynthetic rate
Q_N^{ref}	QN_phy_max maximum N-quota
Q_P^{ref}	QP_phy_max subsistence P-quota
Q_{Si}^{ref}	QSi_phy_max subsistence Si-quota
Q_{N0}	QN_phy_0 subsistence N-quota
Q_{P0}	QP_phy_0 subsistence P-quota
Q_{Si0}	QSi_phy_0 subsistence Si-quota

Chapter 1

MAECS: Model for Adaptive Ecosystems in Coastal Seas

1.1 General Overview

With regard to the conceptualization of trophic interactions, MAECS resembles the classical N-P-Z-D models (1). The emphasis is on the regulation of a number of physiological processes in the phytoplankton unit, which are resolved following a sophisticated optimization scheme. Energy & nutrient allocation scheme of the phytoplankton, as well as the optimization concept used in MAECS is hinted by the model described by [Wirtz and Pahlow \(2010\)](#). See the last section (2) (available only in the pdf form of the documentation) for a detailed description and model equations with some narration.

Todo (1) include a diagram here

In order to re-generate this documentation, make sure the prerequisites (doxygen, latex, bibtex) installed in your system, then type:

```
make
```

inside:

```
$FABMDIR/src/models/hzg/maecs/doc/
```

Chapter 2

Todo List

Type [fabm_hzg_maecs](#)

cross-check the new units and long names of id_Rub (bulk), id_chl (bulk), id_chl2 (diag, chl:c ratio)

Subprogram [fabm_hzg_maecs::maecs_do](#) (self, _ARGUMENTS_DO_)

: min_mass correction of phy% and phy%N at this stage requires specification of threshold values. What about back-calculating phy%reg%N from the smooth_small corrected phy%Q%N?

: specific graz_rate becomes pop. grazing rate. Do this at the rhs calculations

: graz_rate: no temperature modification: forgotten?

: add the rhs equations

: add the rhs equations

page [MAECS: Model for Adaptive Ecosystems in Coastal Seas](#)

(1) include a diagram here

Subprogram [maecs_functions::calc_internal_states](#) (maecs, phy, det, dom, zoo)

the theta-related calculations should obviously be related to Eq. 8.21 but I get lost. See the Q's therein

Subprogram [maecs_functions::calc_sensitivities](#) (maecs, sens, phy, env, nut)

: a more intuitive name like calc_potentials?

: Q: why maecs instead of self?

Subprogram [maecs_functions::min_mass](#) (maecs, phy, method)

: assign some meaningful names to case numbers?

: mm_method to be read from the nml?

: Q: phy%reg%P either non existent or commented out for different cases. Why?

: add equations

Subprogram [maecs_functions::queuederiv](#) (n, x)

: add equations

Subprogram [maecs_functions::queuefunc](#) (n, x, qfunc, qderiv)

: add equations

Subprogram [maecs_grazing::grazing_losses](#) (zoo, resC, Q_pre, lossZNut, lossZDet, mswitch)

: add a description and details

Chapter 3

Data Type Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

fabm_hzg_maecs	9
maecs_functions	13
maecs_grazing	17
maecs_primprod	18
maecs_types	19
maecs_types::stoich_pointer	20
type_base_model	
maecs_types::type_maecs_base_model	21
maecs_types::type_maecs_allocation_fractions	21
type_maecs_base_model	
fabm_hzg_maecs::type_hzg_maecs	20
maecs_types::type_maecs_basic_traits	21
maecs_types::type_maecs_env	21
maecs_types::type_maecs_nutindex	22
maecs_types::type_maecs_om	22
maecs_types::type_maecs_life	22
maecs_types::type_maecs_phy	23
maecs_types::type_maecs_zoo	24
maecs_types::type_maecs_rhs	23
maecs_types::type_maecs_sensitivities	23
maecs_types::type_maecs_switch	23
maecs_types::type_maecs_traitdyn	24

Chapter 4

Data Type Index

4.1 Data Types List

Here are the data types with brief descriptions:

fabm_hzg_maecs	9
This is the module registered in FABM	
maecs_functions	13
Functions called by maecs_do , maecs_grazing and maecs_primprod	
maecs_grazing	17
Grazing module	
maecs_primprod	18
Primary production module	
maecs_types	19
Data types used in fabm_hzg_maecs are defined here	
maecs_types::stoich_pointer	20
fabm_hzg_maecs::type_hzg_maecs	20
This is the model type FABM uses to create the mode	
maecs_types::type_maecs_allocation_fractions	21
maecs_types::type_maecs_base_model	21
maecs_types::type_maecs_basic_traits	21
maecs_types::type_maecs_env	21
maecs_types::type_maecs_life	22
maecs_types::type_maecs_nutindex	22
maecs_types::type_maecs_om	22
maecs_types::type_maecs_phy	23
maecs_types::type_maecs_rhs	23
maecs_types::type_maecs_sensitivities	23
maecs_types::type_maecs_switch	23
maecs_types::type_maecs_traitdyn	24
maecs_types::type_maecs_zoo	24

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

maecs_functions.F90	25
maecs_grazing.F90	25
maecs_maecsd_do_combined.F90	
A temporary file built by combining maecs.F90 and maecs_do.F90 for documentation purposes	26
maecs_primprod.F90	26
maecs_types.F90	
Maecs_types module	26
mainpage.doxygen	??

Chapter 6

Data Type Documentation

6.1 fabm_hzg_maecs Module Reference

This is the module registered in FABM.

Data Types

- type [type_hzg_maecs](#)
this is the model type FABM uses to create the mode

Public Member Functions

- subroutine [initialize](#) (self, configunit)
initializes the model

Private Member Functions

- subroutine [get_light_extinction](#) (self, _ARGUMENTS_GET_EXTINCTION_)
to calculate light extinction when kc changes with depth
- subroutine [maecs_init_stoichvars](#) (self)
a brief description to be added
- subroutine [maecs_do](#) (self, _ARGUMENTS_DO_)
This is the main routine where right-hand-sides are calculated.
- subroutine [maecs_get_vertical_movement](#) (self, _ARGUMENTS_GET_VERTICAL_MOVEMENT_)
handles vertical movement for depth-varying movement rates

6.1.1 Detailed Description

This is the module registered in FABM.

all the [maecs_types](#) are made available to this module

Todo cross-check the new units and long names of id_Rub (bulk), id_chl (bulk), id_chl2 (diag, chl:c ratio)

Definition at line 11 of file maecs_maecsd_do_combined.F90.

6.1.2 Member Function/Subroutine Documentation

6.1.2.1 subroutine fabm_hzg_maecs::initialize (class (type_hzg_maecs), intent(inout), target self, integer, intent(in) configunit)

initializes the model

here the maecs namelists are read and assigned respectively in the model type (self), state & diagnostic variables are registered in FABM and dependencies are imported from FABM

Model parameters, descriptions and corresponding symbols used in formulas:

Parameters

<i>nutN_initial</i>	DIN Dissolved Inorganic Nitrogen DIN
<i>nutP_initial</i>	Phy _P Dissolved Inorganic Phosphorus DIP
<i>nutS_initial</i>	Phy _{Si} Dissolved Inorganic Silicon Si
<i>phyC_initial</i>	Phy _C Phytoplankton Carbon
<i>phyN_initial</i>	phyN _i initial Phytoplankton Nitrogen
<i>phyP_initial</i>	phyP _i initial Phytoplankton Phosphorus
<i>phyS_initial</i>	phyS _i initial Phytoplankton Silicon
<i>zooC_initial</i>	zooC _i initial Zooplankton Carbon
<i>detC_initial</i>	detC _i initial Detritus Carbon
<i>detN_initial</i>	detN _i initial Detritus Nitrogen
<i>detP_initial</i>	detP _i initial Detritus Phosphorus
<i>detS_initial</i>	detS _i initial Detritus Silicon
<i>domC_initial</i>	domC _i initial Dissolved Organic Carbon
<i>domN_initial</i>	domN _i initial Dissolved Organic Nitrogen
<i>domP_initial</i>	domP _i initial Dissolved Organic Phosphorus
<i>frac_Rub_ini</i>	f_R fraction of Rubisco {Rub , f_R {Phy}_{C}}, trait x biomass}
<i>frac_chl_ini</i>	$f_\theta \theta_C$ Chl:C ratio {chl , $f_{\{Phy\}_C}$, trait x biomass}
<i>P_max</i>	P_{max} maximum potential photosynthetic rate
<i>alpha</i>	α specific light adsorption by chloroplasts
<i>sigma</i>	σ Q-dependency of Rubisco activity/chloroplast ratio
<i>theta_LHC</i>	θ_C Chlorophylla-to-C ratio of LHC
<i>rel_chloropl_min</i>	$rel_{chloropl_m}$ in chloroplast-C to phy-C ratio
<i>QN_phy_0</i>	Q_{N0} subsistence N-quota
<i>QN_phy_max</i>	Q_N^{ref} maximum N-quota { V_{NC_max} , $V_{\{max,N\}}^0$, maximum N uptake rate, 0.3 mol--N/(mol-C d)}
<i>AffN</i>	A_N^0 N-Affinity
<i>zeta_CN</i>	ζ_{CN} respiratory costs of N-synthesis/NO ₃ -reduction
<i>zstoich_PN</i>	$zstoich_PN$ P-stoichiometry of active compounds(-> P costs)
<i>exud_phy</i>	$exud_{phy}$ phytoplankton exudation per production
<i>QP_phy_0</i>	Q_{P0} subsistence P-quota
<i>QP_phy_max</i>	Q_P^{ref} subsistence P-quota { V_{PC_max} , $V_{\{max,P\}}^0$, maximum P uptake rate, 0.03 mol--P/(mol-C d)}
<i>AffP</i>	A_P^0 P-Affinity
<i>QSi_phy_0</i>	Q_{Si0} subsistence Si-quota
<i>QSi_phy_max</i>	Q_{Si}^{ref} subsistence Si-quota { V_{SiC_max} , $V_{\{max,Si\}}^0$, maximum Si-uptake rate, 0.02 mol--Si/(mol-C d)}

<i>AffSi</i>	A_{Si}^0 Si-Affinity
<i>syn_nut</i>	$1/h$ synchrony n_queue in nutrient quota limitation
<i>adap_rub</i>	δ_R adap_rub
<i>adap_theta</i>	δ_θ adap_theta
<i>tau_regV</i>	Δt_v tau_regV
<i>phi_agg</i>	ϕ_{agg} quadratic aggregation rate
<i>vS_phy</i>	$v_{S_{phy}}$ sinking velocity for phytoplankton
<i>vS_det</i>	$v_{S_{det}}$ sinking velocity for detritus
<i>hydrol</i>	hydrol hydrolysis rate
<i>remin</i>	remin pelagic remineralisation
<i>Ae_all</i>	$A_{e_{all}}$ Activation energy
<i>T_ref</i>	T_{ref} reference temperature
<i>NutOrder</i>	NutOrder element order of recursive scheme. lower digit: synchrony element
<i>const_NC_zoo</i>	$const_{NC_{zoo}}$ zooplankton N:C ratio
<i>const_PC_zoo</i>	$const_{PC_{zoo}}$ zooplankton P:C ratio
<i>g_max</i>	g_{max} maximum grazing rate
<i>k_grazC</i>	k_{grazC} half saturation graing
<i>yield_zoo</i>	$yield_{zoo}$ yield of herbivory
<i>basal_resp_zoo</i>	$basal_{resp_{zoo}}$ basal respiration
<i>mort_zoo</i>	$mort_{zoo}$ quadratic mortality
<i>a_water</i>	a_{water} background attenuation coefficient
<i>a_spm</i>	a_{spm} attenuation coefficient of SPM
<i>a_chl</i>	a_{chl} attenuation coefficient due to Chl absorption
<i>frac_PAR</i>	$frac_{PAR}$ photosynthetically active fraction of light
<i>small</i>	small lower limit for denominator in ratios; $small_{finite} = \sqrt{small}$
<i>dil</i>	dil dilution of all concentrations except dissolved inorganics

Definition at line 129 of file maecs_maecsdo_combined.F90.

References maecs_init_stoichvars().

6.1.2.2 subroutine fabm_hzg_maecs::get_light_extinction (class(type_hzg_maecs), intent(in) self,
 _ARGUMENTS_GET_EXTINCTION_) [private]

to calculate light extinction when kc changes with depth

extinction coef= $a_{water} + a_{spm} * (p + d + z)$

Definition at line 556 of file maecs_maecsdo_combined.F90.

6.1.2.3 subroutine fabm_hzg_maecs::maecs_init_stoichvars (class(type_hzg_maecs), intent(inout) self)
 [private]

a brief description to be added

some details to be added

Definition at line 588 of file maecs_maecsdo_combined.F90.

Referenced by initialize().

6.1.2.4 fabm_hzg_maecs::maecs_do (class (type_hzg_maecs), intent(in) self, _ARGUMENTS_DO_)
 [private]

This is the main routine where right-hand-sides are calculated.

NOTE: Although this subroutine looks as if it's not a part of any module, it is temporarily included in the [fabm_hzg_maecs](#) module (inside maecs.F90) when compiling the documentation, such that the subroutine is documented under the 'Data Type Documentation' chapter, where the in-body docs are listed

Phytoplankton Equations

We distinguish between mass state variables (in units of carbon, nitrogen, & phosphorus) and property state variables. {For a textual narration and equations, see sec.,sec:ModStr,}

Current 'traits' are:

- nitrogen allocated to rubisco [-] (frac_Rub)
- Chla content of chloroplasts [chl-a/chl-C] (theta)

General code structure:

1. Calculation of quotas, internal states, potential rates
2. Calculation of fluxes, mass exchange rates & rates of change of traits variables
3. Assign mass exchange rates ('rhs(j,i)')
4. Assign rates of change of 'traits' property variables

Detailed Descriptions:

1. Calculation of quotas, internal states, potential rates
 - call min_mass with method=2, store phy%C and %N in phy%reg
 - call calc_internal_states: retrieve phy%Q%X, phy%theta, phy%frac%X
 - if PhotoacclimOn=.false., calculate:
 - phy%chl=phy%C * self%frac_chl_ini
 - phy%frac%theta = self%frac_chl_ini * self%itheta_max
 - phy%theta= self%frac_chl_ini / (self%frac_Rub_ini * phy%relQ%N**self%sigma)
 - call calc_sensitivities: retrieve potential rates: f_T , sens%upt_pot%C (=LH), sens%upt_pot%X (= V_X), sens%P_max
 - Todo** : min_mass correction of phy% and phy%N at this stage requires specification of threshold values. What about back-calculating phy%reg%N from the smooth_small corrected phy%Q%N?
2. Calculation of fluxes, mass exchange rates & rates of change of traits variables & Specify rates of change of traits variables
 - call `maecs_primprod::photosynthesis()`: this is where everything happens!
 - if GrazingOn:
 - graz_rate=rate retrieved from call `maecs_grazing::grazing()`
 - lossZ%X=lossZNut%X, floppZ%X=lossZDet%X retrieved from call `maecs_grazing::grazing_losses()`
 - calculate graz_rate retr= graz_rate * zoo%C and zoo_mort
 - calc. aggreg_rate = $\phi_{agg} * (1 - e^{-0.02 * dom\%C}) * phy\%N * det\%N$
 - future work: aggreg_rate=f(size), (see section 8.2.8)
 - calc. degradT=self%hydrol * f_T and reminT=self%remin * f_T
 - Todo** : specific graz_rate becomes pop. grazing rate. Do this at the rhs calculations
 - : graz_rate: no temperature modification: forgotten?
3. Assign mass exchange rates ('rhs(j,i)')
 - phyC= uptake - dil - exud - aggreg_rate - graz_rate
 - Todo** : add the rhs equations
4. Assign rates of change of 'traits' property variables
 - if PhotoacclimOn: rhsvchl

- if RubiscoOn: rhsvRub

Todo : add the rhs equations

Definition at line 683 of file maecs_maecsdo_combined.F90.

References `maecs_functions::calc_internal_states()`, `maecs_functions::calc_sensitivities()`, `maecs_grazing::grazing()`, `maecs_grazing::grazing_losses()`, `maecs_functions::min_mass()`, and `maecs_primprod::photosynthesis()`.

6.1.2.5 subroutine `fabm_hzg_maecs::maecs_get_vertical_movement` (class (type_hzg_maecs), intent(in) self,
 `_ARGUMENTS_GET_VERTICAL_MOVEMENT_`) [private]

handles vertical movement for depth-varying movement rates

phyto sinking rate depends on the nutritional state, so for each node:

$phy\%relQ$ obtained by calling `calc_internal_states(self,phy,det,dom,zoo)`

then $phyQstat = phy\%relQ\%N * phy\%relQ\%P$

finally, `vsink = maecs_functions::sinking(self%vS_phy, phyQstat, vsink)`

Definition at line 1105 of file maecs_maecsdo_combined.F90.

References `maecs_functions::calc_internal_states()`, `maecs_functions::min_mass()`, and `maecs_functions::sinking()`.

The documentation for this module was generated from the following file:

- `maecs_maecsdo_combined.F90`

6.2 maecs_functions Module Reference

functions called by `maecs_do`, `maecs_grazing` and `maecs_primprod`

Public Member Functions

- subroutine, public `calc_internal_states` (maecs, phy, det, dom, zoo)
calculate the internal states
- subroutine, public `calc_sensitivities` (maecs, sens, phy, env, nut)
calculate sensitivities
- pure real(rk) function, public `uptflex` (Aff0, Vmax0, Nut, fAv)
calc's pot nut upt as f(external conc, allocations)
- pure real(rk) function `foptupt` (Aff0, Vmax0, Nut)
opt. partitioning between surf upt sites and intern. enzymes for nut assim.
- subroutine, public `queuefunc` (n, x, qfunc, qderiv)
the queue function
- subroutine `queuefunc1` (n, x, qfunc, qderiv)
numerical approximation of the queue function
- real(rk) function, public `queuederiv` (n, x)
derivative of the queue function ??
- subroutine, public `sinking` (vS, phys_status, sinkvel)
calculation of the sinking rate
- subroutine, public `min_mass` (maecs, phy, method)

minimum mass

- pure real(rk) function, public [smooth_small](#) (x, eps)

continuous smoothing function by kw Apr 2012

6.2.1 Detailed Description

functions called by `maecs_do`, [maecs_grazing](#) and [maecs_primprod](#)

Definition at line 8 of file `maecs_functions.F90`.

6.2.2 Member Function/Subroutine Documentation

6.2.2.1 `maecs_functions::calc_internal_states (class (type_maecs_base_model), intent(in) maecs, type (type_maecs_phy), intent(inout) phy, type (type_maecs_om), intent(inout) det, type (type_maecs_om), intent(inout) dom, type (type_maecs_zoo), intent(inout) zoo)`

calculate the internal states

Todo the theta-related calculations should obviously be related to Eq. 8.21 but I get lost. See the Q's therein

1. Calculate elemental absolute and relative quotas (Q and relQ):

- $\text{phy}\%Q\%X = \text{phy}\%X / \text{phy}\%C$ where $x=N,P,Si$
- $\text{phy}\%relQ\%X = (\text{phy}\%Q\%X - \text{maecs}\%qN_phy_0) / \text{maecs}\%iK_QN$ where $x=N,P,Si$

2. Calculate Rubisco fraction (convert from the bulk variable)

- unpack $\text{phy}\%frac\%Rub (= f_R) = \text{phy}\%Rub / \text{phy}\%reg\%C$
 - smooth $1 - f_R$ to $(\text{small}\%finite + \text{rel_chlropl_min})$ (both nml pars), such that f_R is always smaller than 1
 - $f_R = \text{phy}\%Rub / \text{phy}\%C$ - note the $1 - (1 - x)$ structure
 - which equals x for small x, and is slightly below x for $x > 1$
 - a counterpart of smooth-small, could be called smooth-at-one;
 - ensures that f_R is always somewhat smaller than one, since it
 - leaves a minimal fraction of resources to other compartments (f_V)
3. Calculate θ and f_θ See also: [8.2.4](#)

- $\text{phy}\%rel_chloropl = f_R * q_N^C$
 - $\text{phy}\%rel_chloropl$ = factor that relates "chl-a per chloroplast" to "chl-a per cell-C" thus chloroplast-C over total intracellular C
- $\text{phy}\%\theta = \text{phy}_{chl} / \text{phy}_C / \text{phy}\%rel_chloropl$
 - eq: f_θ says $\text{phy}_{chl} / \text{phy}_C = f_\theta / \theta_C$
 - $\text{phy}\%chl$: bulk variable. biomass (phyC) times trait (theta) times factor)
 - this choice converts a "bulk trait ($\theta \text{ phy}_C$) to a observable, i.e. bulk CHL-a conc.
- $\text{phy}\%frac\%\theta = \text{phy}_{chl} / \text{phy}_C * \text{maecs}\%i\theta_{max}$
- $f_V = \text{phy}\%frac\%\text{TotFree} - f_\theta - f_R$, where $\text{phy}\%frac\%\text{TotFree} = 1.0$

3. Calculate zooplankton states:

- $zoo_{QX} = \text{maecs}\%const_N C_{zoo}, zoo_X = zoo_C * zoo_{QN}, X = N, P$
- $zoo_{yield} = \text{maecs}\%yield_{zoo}, zoo_{flopp} = 1 - \text{maecs}\%yield_{zoo}$

Definition at line 27 of file maecs_functions.F90.

References smooth_small().

Referenced by fabm_hzg_maecs::maecs_do(), and fabm_hzg_maecs::maecs_get_vertical_movement().

6.2.2.2 subroutine, public maecs_functions::calc_sensitivities (class (type_maecs_base_model), intent(in) *maecs*, type (type_maecs_sensitivities), intent(out) *sens*, type (type_maecs_phy), intent(in) *phy*, type (type_maecs_env), intent(in) *env*, type (type_maecs_om), intent(in) *nut*)

calculate sensitivities

Details:

- sens%f_T see eq. 8.13
- sens%P_max see eq. 8.20
- sens%upt_pot%C (=LH) see eq. 8.11
- sens%upt_pot%X, (= V_X), X=N,P,Si calculated by `uptflex()` according to eq. 8.26

Todo : a more intuitive name like calc_potentials?

: Q: why maecs instead of self?

Definition at line 167 of file maecs_functions.F90.

References fopty(), smooth_small(), and uptflex().

Referenced by fabm_hzg_maecs::maecs_do().

6.2.2.3 pure real(rk) function, public maecs_functions::uptflex (real(rk), intent(in) *Aff0*, real(rk), intent(in) *Vmax0*, real(rk), intent(in) *Nut*, real(rk), intent(in) *fAv*)

calc's pot nut upt as f(external conc, allocations)

uptake regulation: sites vs. processing

see eq. 8.23 - 8.29

- $\text{sensupt_potX} = (A_X * DIX * V_{max,X}) / (A_X * DIX + V_{max,X})^{-1}$
- $A_X = f_{A,X} * A_X^0$, where $f_{A,X} = \text{fOptUpt}()$
- $V_{max} = (1 - f_{A,X}) * V_{max}^0 * f_T$

Definition at line 233 of file maecs_functions.F90.

Referenced by calc_sensitivities().

6.2.2.4 pure real(rk) function maecs_functions::fopty (real(rk), intent(in) *Aff0*, real(rk), intent(in) *Vmax0*, real(rk), intent(in) *Nut*)

opt. partitioning between surf upt sites and intern. enzymes for nut assim.

calculates $f_{A,X}$ see eq. 8.29.

Definition at line 249 of file maecs_functions.F90.

Referenced by calc_sensitivities().

6.2.2.5 subroutine, public `maecs_functions::queuefunc` (`real(rk)`, intent(in) *n*, `real(rk)`, intent(in) *x*, `real(rk)`, intent(out) *qfunc*, `real(rk)`, intent(out) *qderiv*)

the queue function

provides both the queuing function and it's derivative with the parameter $n \rightarrow \infty$:liebig and $n \sim 1$:product see: Section [8.2.3](#)

Todo : add equations

Definition at line 264 of file `maecs_functions.F90`.

Referenced by `maecs_primprod::photosynthesis()`.

6.2.2.6 subroutine `maecs_functions::queuefunc1` (`real(rk)`, intent(in) *n*, `real(rk)`, intent(in) *x*, `real(rk)`, intent(out) *qfunc*, `real(rk)`, intent(out) *qderiv*)

numerical approximation of the queue function

$n \rightarrow \infty$:liebig $n \sim 1$:product

Here is an example of adding a snip of code:

```
nn = n+1.
hh = 1./nn
x0 = (log(exp(nn)-1))*hh
en = exp(-nn*(x/(1+hh*x) - x0))
qfunc = 1. - hh*log(1.+ en)
qderiv = 1. / ( (1. + hh * x)**2 * (1.+1./en))
```

Definition at line 287 of file `maecs_functions.F90`.

6.2.2.7 `real(rk)` function, public `maecs_functions::queuederiv` (`real(rk)`, intent(in) *n*, `real(rk)`, intent(in) *x*)

derivative of the queue function ??

Returns

`queuederiv`

Todo : add equations

Definition at line 310 of file `maecs_functions.F90`.

6.2.2.8 subroutine, public `maecs_functions::sinking` (`real(rk)`, intent(in) *vS*, `real(rk)`, intent(in) *phys_status*, `real(rk)`, intent(out) *sinkvel*)

calculation of the sinking rate

- $\text{sinkvel} = -vS * e^{-4 * \text{phys_status}}$
 - *phys_status* calculated in `fabm_hzg_maecs::maecs_get_vertical_movement()`
 - Future work: $\text{sinkvel} = f(\text{size})$ (see section [8.2.7](#))

Definition at line 327 of file `maecs_functions.F90`.

Referenced by `fabm_hzg_maecs::maecs_get_vertical_movement()`.

6.2.2.9 subroutine, public `maecs_functions::min_mass` (`class` (`type_maecs_base_model`), intent(in) *maecs*, `type` (`type_maecs_phy`), intent(inout) *phy*, `integer`, intent(in), optional *method*)

minimum mass

pushes the *phyC*, *N* and *P* to some lower boundary according to 4 different methods (controlled by the *mm_method* parameter): *phy%N* and *phy%C* are stored in *phy%reg%N* and *phy%reg%C*, respectively

1. if phy%N <= 1e-7; phy%N=1e-7, phy%C=phy%N/QN(aver), phy%P=phy%C*QP(aver)
2. ..
3. ..
4. ..

Todo : assign some meaningful names to case numbers?

: mm_method to be read from the nml?

: Q: phy%reg%P either non existent or commented out for different cases. Why?

: add equations

Definition at line 397 of file maecs_functions.F90.

References smooth_small().

Referenced by fabm_hzg_maecs::maecs_do(), and fabm_hzg_maecs::maecs_get_vertical_movement().

6.2.2.10 pure real(rk) function, public maecs_functions::smooth_small (real(rk), intent(in) x, real(rk), intent(in) eps)

continuous smoothing function by kw Apr 2012

smoothly converges x to **eps/2** for x<eps

$$x = eps + (x - eps) * e^{x/eps} / (1 + e^{x/eps})$$

Definition at line 504 of file maecs_functions.F90.

Referenced by calc_internal_states(), calc_sensitivities(), min_mass(), and maecs_primprod::photosynthesis().

The documentation for this module was generated from the following file:

- [maecs_functions.F90](#)

6.3 maecs_grazing Module Reference

Grazing module.

Public Member Functions

- subroutine, public [grazing](#) (Imax, HalfSat, preyconc, rate)
calculates grazing rate
- subroutine, public [grazing_losses](#) (zoo, resC, Q_pre, lossZNut, lossZDet, mswitch)
loss rates of grazers to inorganic and organic particulate

6.3.1 Detailed Description

Grazing module.

Definition at line 7 of file maecs_grazing.F90.

6.3.2 Member Function/Subroutine Documentation

6.3.2.1 subroutine, public `maecs_grazing::grazing` (`real(rk)`, intent(in) *lmax*, `real(rk)`, intent(in) *HalfSat*, `real(rk)`, intent(in) *preyconc*, `real(rk)`, intent(out) *rate*)

calculates grazing rate

$$\text{rate} = I_{\max} * F^2 / (K^2 + F^2)$$

Definition at line 20 of file `maecs_grazing.F90`.

Referenced by `fabm_hzg_maecs::maecs_do()`.

6.3.2.2 subroutine, public `maecs_grazing::grazing_losses` (`type`(`type_maecs_zoo`), intent(in) *zoo*, `real(rk)`, intent(in) *resC*, `type`(`type_maecs_om`), intent(in) *Q_prey*, `type`(`type_maecs_om`), intent(out) *lossZNut*, `type`(`type_maecs_om`), intent(out) *lossZDet*, `type`(`type_maecs_switch`), intent(in) *mswitch*)

loss rates of grazers to inorganic and organic particulate

assumes a constant C:N:P unit feeding on variable C:N:P food

Todo : add a description and details

Definition at line 37 of file `maecs_grazing.F90`.

Referenced by `fabm_hzg_maecs::maecs_do()`.

The documentation for this module was generated from the following file:

- [maecs_grazing.F90](#)

6.4 maecs_primprod Module Reference

Primary production module.

Public Member Functions

- subroutine, public [photosynthesis](#) (`self`, `sens`, `phy`, `uptake`, `exud`, `acc`)
calculates grazing rate

6.4.1 Detailed Description

Primary production module.

Definition at line 7 of file `maecs_primprod.F90`.

6.4.2 Member Function/Subroutine Documentation

6.4.2.1 `maecs_primprod::photosynthesis` (`class`(`type_maecs_base_model`), intent(in) *self*, `type`(`type_maecs_sensitivities`), intent(in), target *sens*, `type`(`type_maecs_phy`), intent(inout) *phy*, `type`(`type_maecs_om`), intent(out), target *uptake*, `type`(`type_maecs_om`), intent(out) *exud*, `type`(`type_maecs_traitdyn`), intent(out), target *acc*)

calculates grazing rate

This is the subroutine, where the optimal regulation of phytoplankton traits are described, which is central to the physiological-MAECS

1. Prepare loop over structure elements by assigning a poinzer structure

- Here, every possible nutrient is asked explicitly; thus first Si then P
- This is an example of adding a whole code source:

```

! nitrogen
  i = self%nutind%iN
  elem(i)%relQ      = smooth_small(phy%relQ%N,eps)
  elem(i)%Q         = smooth_small(phy%Q%N,eps)
  elem(i)%upt_pot   => sens%upt_pot%N
  elem(i)%upt_act   => upt_act%N
  elem(i)%upt       => uptake%N
  elem(i)%iKQ       = self%iK_QN
  elem(i)%aV        => acc%aV%N
  zeta_x(i)         = self%zeta_CN

! phosphorus
if (self%PhosphorusOn) then
  i = self%nutind%iP
  elem(i)%relQ      = smooth_small(phy%relQ%P,eps)
  elem(i)%Q         = smooth_small(phy%Q%P,eps)
  elem(i)%upt_pot   => sens%upt_pot%P
  elem(i)%upt_act   => upt_act%P
  elem(i)%upt       => uptake%P
  elem(i)%iKQ       = self%iK_QP
  elem(i)%aV        => acc%aV%P
  zeta_x(i)         = self%zeta_CN * ((1.-f_lip)*q_nolip + f_lip*q_lip)
end if

! silicon
if (self%SiliconOn) then
  i = self%nutind%iSi
  elem(i)%relQ      = smooth_small(phy%relQ%Si,eps)
  elem(i)%Q         = smooth_small(phy%Q%Si,eps)
  elem(i)%upt_pot   => sens%upt_pot%Si
  elem(i)%upt_act   => upt_act%Si
  elem(i)%upt       => uptake%Si
  elem(i)%iKQ       = self%iK_QSi
  elem(i)%aV        => acc%aV%Si
  zeta_x(i)         = 0.0_rk
end if
,

```

Definition at line 20 of file `maecs_primprod.F90`.

References `maecs_functions::queuefunc()`, and `maecs_functions::smooth_small()`.

Referenced by `fabm_hzg_maecs::maecs_do()`.

The documentation for this module was generated from the following file:

- [maecs_primprod.F90](#)

6.5 maecs_types Module Reference

Data types used in `fabm_hzg_maecs` are defined here.

Data Types

- type [stoich_pointer](#)
- type [type_maecs_allocation_fractions](#)
- type [type_maecs_base_model](#)
- type [type_maecs_basic_traits](#)
- type [type_maecs_env](#)
- type [type_maecs_life](#)
- type [type_maecs_nutindex](#)
- type [type_maecs_om](#)
- type [type_maecs_phy](#)
- type [type_maecs_rhs](#)

- type [type_maecs_sensitivities](#)
- type [type_maecs_switch](#)
- type [type_maecs_traitdyn](#)
- type [type_maecs_zoo](#)

6.5.1 Detailed Description

Data types used in [fabm_hzg_maecs](#) are defined here.

todo: we should consider including parameter tables for some maecs types. Parser could do it just like in the initialize subroutine.

Definition at line 14 of file [maecs_types.F90](#).

The documentation for this module was generated from the following file:

- [maecs_types.F90](#)

6.6 [maecs_types::stoich_pointer](#) Type Reference

6.6.1 Detailed Description

Definition at line 121 of file [maecs_types.F90](#).

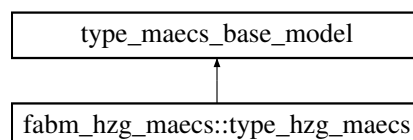
The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.7 [fabm_hzg_maecs::type_hzg_maecs](#) Type Reference

this is the model type FABM uses to create the mode

Inheritance diagram for [fabm_hzg_maecs::type_hzg_maecs](#):



6.7.1 Detailed Description

this is the model type FABM uses to create the mode

the parent type ([type_maecs_base_model](#)) was defined in [maecs_types](#) module

Definition at line 29 of file [maecs_maecsdo_combined.F90](#).

The documentation for this type was generated from the following file:

- [maecs_maecsdo_combined.F90](#)

6.8 maecs_types::type_maecs_allocation_fractions Type Reference

6.8.1 Detailed Description

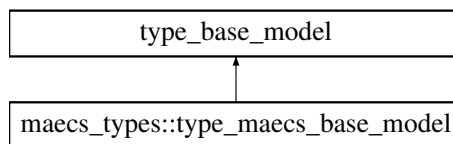
Definition at line 71 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.9 maecs_types::type_maecs_base_model Type Reference

Inheritance diagram for maecs_types::type_maecs_base_model:



6.9.1 Detailed Description

Definition at line 27 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.10 maecs_types::type_maecs_basic_traits Type Reference

6.10.1 Detailed Description

Definition at line 66 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.11 maecs_types::type_maecs_env Type Reference

6.11.1 Detailed Description

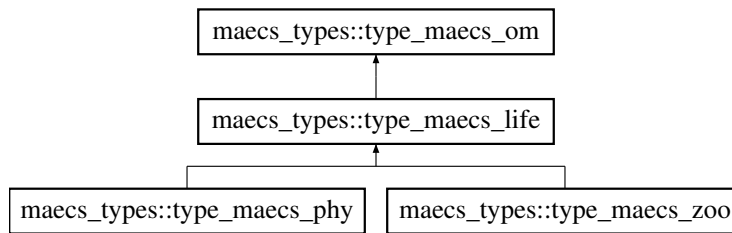
Definition at line 47 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.12 maecs_types::type_maecs_life Type Reference

Inheritance diagram for maecs_types::type_maecs_life:



6.12.1 Detailed Description

Definition at line 78 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.13 maecs_types::type_maecs_nutindex Type Reference

6.13.1 Detailed Description

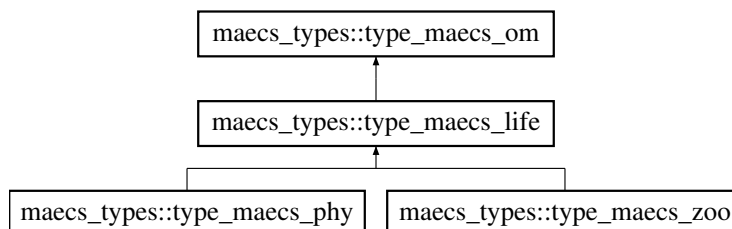
Definition at line 22 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.14 maecs_types::type_maecs_om Type Reference

Inheritance diagram for maecs_types::type_maecs_om:



6.14.1 Detailed Description

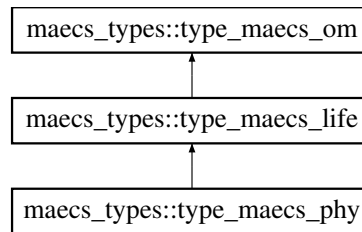
Definition at line 62 of file maecs_types.F90.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.15 `maecs_types::type_maecs_phy` Type Reference

Inheritance diagram for `maecs_types::type_maecs_phy`:



6.15.1 Detailed Description

Definition at line 83 of file `maecs_types.F90`.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.16 `maecs_types::type_maecs_rhs` Type Reference

6.16.1 Detailed Description

Definition at line 50 of file `maecs_types.F90`.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.17 `maecs_types::type_maecs_sensitivities` Type Reference

6.17.1 Detailed Description

Definition at line 112 of file `maecs_types.F90`.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.18 `maecs_types::type_maecs_switch` Type Reference

6.18.1 Detailed Description

Definition at line 57 of file `maecs_types.F90`.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.19 `maecs_types::type_maecs_traitdyn` Type Reference

6.19.1 Detailed Description

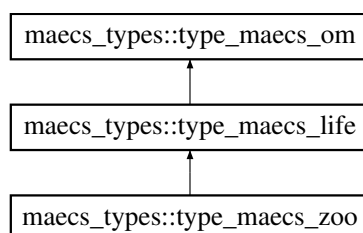
Definition at line 100 of file `maecs_types.F90`.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

6.20 `maecs_types::type_maecs_zoo` Type Reference

Inheritance diagram for `maecs_types::type_maecs_zoo`:



6.20.1 Detailed Description

Definition at line 94 of file `maecs_types.F90`.

The documentation for this type was generated from the following file:

- [maecs_types.F90](#)

Chapter 7

File Documentation

7.1 `maecs_functions.F90` File Reference

```
#include "fabm_driver.h"
```

Data Types

- module `maecs_functions`
functions called by `maecs_do`, `maecs_grazing` and `maecs_primprod`

7.1.1 Detailed Description

Author

Richard Hofmeister, Markus Schartau, Kai Wirtz, Onur Kerimoglu

Definition in file `maecs_functions.F90`.

7.2 `maecs_grazing.F90` File Reference

```
#include "fabm_driver.h"
```

Data Types

- module `maecs_grazing`
Grazing module.

7.2.1 Detailed Description

Author

Richard Hofmeister, Markus Schartau, Kai Wirtz

Definition in file `maecs_grazing.F90`.

7.3 maecs_maecsdo_combined.F90 File Reference

A temporary file built by combining maecs.F90 and maecs_do.F90 for documentation purposes.

```
#include "fabm_driver.h"  
#include "maecs_do.F90"
```

Data Types

- module [fabm_hzg_maecs](#)
This is the module registered in FABM.
- type [fabm_hzg_maecs::type_hzg_maecs](#)
this is the model type FABM uses to create the mode

7.3.1 Detailed Description

A temporary file built by combining maecs.F90 and maecs_do.F90 for documentation purposes.

Definition in file [maecs_maecsdo_combined.F90](#).

7.4 maecs_primprod.F90 File Reference

```
#include "fabm_driver.h"
```

Data Types

- module [maecs_primprod](#)
Primary production module.

7.4.1 Detailed Description

Author

Kai Wirtz, Richard Hofmeister, Markus Schartau, Onur Kerimoglu

Definition in file [maecs_primprod.F90](#).

7.5 maecs_types.F90 File Reference

[maecs_types](#) module

```
#include "fabm_driver.h"
```

Data Types

- module `maecs_types`

Data types used in `fabm_hzg_maecs` are defined here.

- type `maecs_types::type_maecs_nutindex`
- type `maecs_types::type_maecs_base_model`
- type `maecs_types::type_maecs_env`
- type `maecs_types::type_maecs_rhs`
- type `maecs_types::type_maecs_switch`
- type `maecs_types::type_maecs_om`
- type `maecs_types::type_maecs_basic_traits`
- type `maecs_types::type_maecs_allocation_fractions`
- type `maecs_types::type_maecs_life`
- type `maecs_types::type_maecs_phy`
- type `maecs_types::type_maecs_zoo`
- type `maecs_types::type_maecs_traitdyn`
- type `maecs_types::type_maecs_sensitivities`
- type `maecs_types::stoich_pointer`

7.5.1 Detailed Description

`maecs_types` module

Author

Richard Hofmeister, Kai Wirtz

Definition in file `maecs_types.F90`.

Chapter 8

Textual Model Description

8.1 Model structure

MAECS resolves major functional groups and their dynamics not unlike simple state-of-the-art ecosystem models. All energy and material fluxes in the ecosystem derive from primary production of phytoplankton, here expressed in terms of biomass carbon (C) concentration. Phytoplankton experiences various local and trait dependent loss rates, from sinking, respiration, exudation, to grazing. MAECS versions differ in the degree of resolution in ecological processes; a full account of grazing interactions in the plankton is only addressed by versions that includes size as a major trait. Grazers convert only a fraction of captured phytoplankton to their own biomass; in size-based variants of MAECS we have implemented different formulations for whether this fraction immediately contributes to the biomass of adult grazers or their egg stage, or how many grazer groups are simulated (see below).

Physiological regulation in unicellular autotrophs makes the very kernel of the MAECS version documented here. The intracellular nutrient quotas, in this model version resolved for nitrogen (N), phosphorus (P), and silicon (Si), change over time when uptake does not match demand due to biomass build-up. Differential uptake regulation then leads to highly variable stoichiometries, that not only reflect ambient nutrient and light concentration, but also generalized optimality criteria.

For describing the living compartment of plankton ecosystems MAECS employs much more trait variables (e.g., allocation coefficients) compared to bulk variables (e.g., phytoplankton or zooplankton biomass). The adaptive trait dynamics infers the need of calculating (sometimes long) derivative terms, which also challenges this documentation; however, the gradient-derived terms include very few additional process parameters, so that the number of tunable parameters in relation to simulated and testable dynamics is significantly reduced compared to models that share a similar resolution in plankton physiology or ecology. Most importantly, MAECS seeks to account for biophysical and evolutionary principles as much as possible; resulting in expressions replacing heuristic functions such as Michaelis-Menten approximations of nutrient uptake, Droop representation of cell growth, or Liebig's rule of minimum. Biophysical origins of model formulations further reduces the number of tunable parameters, in particular in the ecological part. Prominent counter-examples, i.e. yet uncertain model formulations or parameters, comprise aggregation dynamics or variations in metabolic interdependency.

8.1.1 Mass equations

Basic mass equations common to the two MAECS variants (size and physiology). All coefficients and their meanings are listed in Table ??.

Net change in autotroph C, N, P, Si (cf. Eq.(8.8) and Eq.(8.31)) ...

$$\frac{d}{dt}\text{Phy}_C = \mu_{\text{tot}} \cdot \text{Phy}_C \quad (8.1)$$

$$\frac{d}{dt}\text{Phy}_X = V_X \cdot \text{Phy}_C - M \cdot \text{Phy}_X \quad \text{with } X = \text{N, P, Si} \quad (8.2)$$

... in dissolved inorganic nutrients ($X=N, P$), optionally including silicate

$$\frac{d}{dt}DIX = -V_X \text{Phy}_C + \omega_{\text{DOM}}^{(8.7)} DOX \quad (8.3)$$

$$\frac{d}{dt}DSi = -V_{Si} \text{Phy}_C + \omega_{\text{Det}}^{(8.7)} \text{Det}_{Si} \quad (8.4)$$

[TODO: include DIC into code]

... in dissolved organic nutrients ($X=C, N, P$),

$$\frac{d}{dt}DOX = E \text{Phy}_X + \omega_{\text{Det}} \text{Det}_X - \omega_{\text{DOM}} DOX \quad (8.5)$$

... and in the detrital pool ($X=C, N, P, Si$).

$$\frac{d}{dt}\text{Det}_X = M' \text{Phy}_X + M_Z \text{Zoo}_X - \omega_{\text{Det}} \text{Det}_X \quad (8.6)$$

where the phytoplankton mortality M' collects contributions from sloppy grazing $((1-y)G)$ and, in 0D setups, and from sinking (S in Eq.(8.35)). Remineralization of DOM ω_{DOM} and hydrolysis of detritus ω_{Det} both change with ambient temperature and substrate quality

$$\omega_{\text{DOM}} = f_T \frac{\text{DON}}{\text{DOC}} \omega_{\text{DOM}}^* \quad \omega_{\text{Det}} = f_T \frac{\text{Det}_N}{\text{Det}_C} \omega_{\text{Det}}^* \quad (8.7)$$

Because of the quality dependency in degradation rates [TODO: justify], element cycles in the water column are easily decoupled within MAECS.

Note that in a chemostat mode, to the dynamics of *all* concentration variables a dilution (D) loss is added, which only for dissolved nutrients also contains reservoir inflow $(-D \cdot (DIX - DIX^0))$.

[TODO: Parameter table]

8.2 Process Descriptions

8.2.1 Growth rate components and primary production

Temporal changes of the bulk phytoplankton concentration Phy_C are per construction given in terms of the over all relative growth rate μ_{tot} :

$$\frac{d}{dt}\text{Phy}_C = \mu_{\text{tot}} \cdot \text{Phy}_C \quad (8.8)$$

The relative growth rate of autotrophic unicells (phytoplankton) collects primary production and a number of loss terms

$$\mu_{\text{tot}} = \underbrace{P - \underbrace{R}_{(8.12)}}_{\mu} - \underbrace{\left(\underbrace{E}_{(8.45)} + \underbrace{\frac{G \text{Zoo}_C}{\text{Phy}_C}}_{(8.34)} + \underbrace{S}_{(8.35)} + \underbrace{AF}_{(8.40)} + D \right)}_M = \mu - M \quad (8.9)$$

with sinking loss given in Eq.(8.35) (relevant only in 0D), aggregate formation in Eq.(8.40) [TODO: living cells in aggregates], grazing G explained in Sec. 8.2.6, exudation rate E formulated in Eq.(8.45), dilution rate D (for running the model in a chemostat mode), and respiration R in Eq.(8.12).

Gross C assimilation P changes with stoichiometric balance as expressed by the multi-nutrient co-limitation factor LF (Eq.(8.14)), maximal photosynthetic capacity P_{max} (further depending on, e.g. partitioning, Eq.(8.20), or size in the ecological version, Wirtz (2011, 2013)) and light harvesting success LH

$$P = \underbrace{P_{\text{max}}}_{(8.20)} \cdot \underbrace{LF}_{(8.14)} \cdot LH \quad (8.10)$$

Light harvesting LH is primarily controlled by ambient light intensity PAR and light adsorption by chloroplasts $\alpha \cdot \theta$ proportional to the chlorophyll concentration θ (cf. Eq.(8.21)). Its functional form derives from Poisson arrival statistics of photons

$$LH = 1 - e^{-\alpha \theta PAR / P_{\max}} \quad (8.11)$$

8.2.2 Respiration and temperature

Respiratory losses reflect costs of N uptake (Raven, 1984; Pahlow, 2005), while neglecting energetic costs of P- and Si-assimilation

$$R = \zeta V_N \quad (8.12)$$

Temperature dependency of respiration thus follows from the factor f_T in the uptake coefficients in Eq.(8.24). In the current MAECS version, all metabolic rates equally increase with rising temperature following the Arrhenius equation,

$$f_T = e^{-E_a(T^{-1} - T_0^{-1})} \quad \text{with} \quad E_a = \frac{T_0^2}{10} \cdot \log(Q_{10}) \quad (8.13)$$

with a reference temperature T_0 (here 18°C) where f_T equals one.

8.2.3 Multi-nutrient co-limitation

The co-limitation factor LF is constructed within a metabolic network view of cellular physiology. There, LF reflects not only the availability of singular resources (nutrients) but also how dependent the sub-networks (protein turnover) associated to those nutrients can evolve. In this view, LF quantifies the turnover of the first sub-process (here intracellular nitrogen turnover) times the queuing function that gives the intermittency between the first and the other sub-processes (associated to elements X):

$$LF = q_N \cdot \underset{(8.16)}{g_h} (q'_X / q_N) \cdot c_{hq} \quad \text{with} \quad c_{hq} = 1 + \underset{(8.17)}{c_h} + h \cdot q_N \underset{(8.18)}{q'_X} \quad (8.14)$$

The relative quota q_X expresses the availability of nutrient X above a subsistence threshold Q_X^0 normalized by a reference pool size $Q_X^* - Q_X^0$. In contrast to most other models resolving variable internal stores (e.g. Morel, 1987), Q_X^* does not impose [TODO: 'does not impose'=? is it rather 'is not a prescribed'? but what is it then?] an upper boundary of the cell quota Q_X .

$$q_X = \frac{Q_X - Q_X^0}{\Delta Q_X} \quad \text{with} \quad \Delta Q_X = Q_X^* - Q_X^0 \quad \text{and} \quad X = N, P, Si, \dots \quad (8.15)$$

In Eq.(8.14), the product $h q_N q'_X$ describes the non-linear metabolic interdependency between intracellular turnover of element N and X that is not covered by the simple queuing function. g_h expresses a "stop and go" dependency while neglecting possible amplification and inhibition effects as explained by Wirtz&Kerimoglu (in prep.). The product term leads to a quadratic influence of the first element (q_N) and can be considered as most simple account of non-linear metabolic interdependency. It also ensures symmetry of LF with respect to exchanging the order of nutrients as $q_1 g_h(q'_2 / q_1) c_{hq12} \approx q_2 g_h(q'_1 / q_2) c_{hq21}$.

The queuing function g_h can be derived from assuming Poisson statistics in phase-locking of sub-networks [TODO: 'Poisson statistics in phase-locking of sub-networks' sounds scary, please expand!]:

$$g_h(x) = \frac{x - x_h}{1 - x_h} \quad \text{with} \quad x_h = x^{1+h^{-1}} \quad (8.16)$$

The function g_h introduces as new control parameter the metabolic interdependency h . h resembles the processing intermittency introduced by Wirtz (2012), which describes the probability of phase-locking [TODO: 'phase-locking' is again an unusual term: expand with a sentence?] in independent sub-steps in a process chain.

The correction coefficient c_h in Eq.(8.14) follows from imposing convergence of LF to the product and Liebig rules. Compliance to the Liebig rule leads to $c_0 = 0$ since g_0 describes a stepwise linear function [TODO: it's not clear: when $ch=h=0$, Eq.(8.14) reduces to $q_N g_0(q'_X/q_N)$, with $g_0 = -\inf / -\inf$]. For the product rule, we use the identity $g_h(1) = 1/(1+h)$ and assume that $q_X = q_Y = 1/2$, to obtain $1/4 = 1/2 \cdot (1 + h/4 + c_h)/(1+h)$ [TODO: it's very unclear where this equation came from], or $c_h = -1/2 + h/4$. However, the offset in this linear relation conflicts with the condition $c_0 = 0$. Both conditions can be approximately reconciled by a logarithmic function [TODO: does Eq.(8.17) have something to do with all the other equations in this paragraph? if so, how? if not, why not directly listing those 'conditions', providing Eq.(8.17) saying that it is one of the potentially many pragmatic solutions?]

$$c_h = \log(1/4^h + h/2) \quad (8.17)$$

It is mandatory in MAECS to resolve nitrogen. If optionally one, two, or more further nutrients are considered (e.g., P or Si, P and Si, or a micro-nutrient), a recursive scheme is applied. The limiting effect of element $X+1$ on the processing of element X (q'_X) is then formally equivalent to the LF of the first element (N) limited by the remainder metabolism (LF = q'_N) as given in Eq.(8.14):

$$q'_X = q_X \cdot g_h(q'_{X+1}/q_X) \cdot c_{hq} \quad X = P, Si, \dots \quad (8.18)$$

For the final element we have $q'_X = q_X$. From the (subjective) ordering of elements in the recursive scheme only a small asymmetry arises. [TODO: no matter how small it is, the asymmetry probably requires us to document (and maybe also provide justification, if any) the order we chose]

8.2.4 Uptake system allocation and PI-coefficients

It is assumed that the fraction of free proteins or allocatable resources does not change. [TODO: it is not clear what this means: maybe a diagram helps?] Structural compounds such as cell wall, nucleus, or other non directly functional components are thus kept at a fixed ratio. The pool of free organics (in terms of cellular carbon) is partitioned between photosynthetic machinery and nutrient uptake. Photosynthetic machinery is further sub-divided into light harvesting and processing. The relative pool size of free resource (here in units C!) invested into light-independent reactions, primarily those in Rubisco and the Calvin cycle (Friend, 1991) is denoted as f_R , the one for the light harvesting complex (LHC) f_θ . This way, the coefficient f_V for C-allocation to nutrient uptake becomes a linear function of the C-allocation to LHC f_θ and to Rubisco f_R :

$$f_V = 1 - f_R - f_\theta \quad (8.19)$$

Maximum photosynthesis rate P_{\max} is controlled by the pool fraction f_R invested into Rubisco/processing and a temperature dependency f_T given in Eq.(8.13), while the effect of nutrient limitation (namely N) is already included in Eq.(8.10)

$$P_{\max} = f_R \cdot \underset{(8.13)}{f_T} \cdot P_{\max}^* \quad (8.20)$$

The fraction of light harvesting carbon depends on the C-fraction partitioned to photosystem processing (electron chain) and the chloroplast chlorophyll concentration θ (relative to a reference value θ_C):

$$f_\theta = f_R q_N^\sigma \theta / \theta_C \quad \text{or} \quad \theta = f_\theta / f_R q_N^{-\sigma} \theta_C \quad (8.21)$$

where C stoichiometry of pigment complexes is denoted by θ_C . The exponent σ describes an additional linkage of LHC synthesis on the N-status of the cell. Wirtz and Pahlow (2010) proposed that $\sigma = 1$ for diatoms, while $\sigma = 0$ for other autotrophs.

[TODO: 1) so far, it's not clear what's the relevance of θ or f_θ for physiological processes. Maybe a sentence. 2) it's also not clear where these novel and mysterious parameters such as f_R and θ come from. Maybe it's helpful to hint that they are being dynamically optimized, referring to Sec. 8.3]

8.2.5 Nutrient uptake

Nutrient uptake rate of the cell varies with a number of internal, physiological factors, such as the partitioning coefficient f_V (see above) and the (enzymatic) activity $a_{V,X}$ ($X = N, P, Si, \dots$), and is furthermore determined by the potential uptake V_X^*

$$V_X = \underset{(8.19)}{f_V} \cdot \underset{(8.61)}{a_{V,X}} \cdot V_X^* \quad (8.22)$$

Potential uptake V_X^* reflects ambient nutrient concentration, but also affinity and transport capacities denoted by A_X and $V_{\max,X}$. Nutrient uptake is a sequential 2-stage process (membrane uptake and intracellular transport) as described by the nutrient affinity A_X and maximal uptake rate $V_{\max,X}$ such that the effective uptake time equals the sum of turnover times in each stage:

$$V_X^{*-1} = V_{\max,X}^{-1} + (A_X \text{ DIX})^{-1} \quad \text{with} \quad \text{DIX} = \text{DIN}, \text{DIP}, \text{DSi}, \dots \quad (8.23)$$

Nutrient uptake characteristics are temperature sensitive and assumed to depend on a sub-partitioning of allocatable proteins expressed by the coefficient $f_{A,X}$

$$V_{\max,X} = (1 - f_{A,X}) \cdot Q_0 \cdot \underset{(8.13)}{f_T} \cdot V_{\max,X}^0 \quad (8.24)$$

$$A_X = f_{A,X} \cdot Q_0 \cdot f_T \cdot A_X^0 \quad (8.25)$$

ONUR: What's presently implemented is:

$$V_{\max,X} = (1 - f_{A,X}) \cdot \underset{(8.13)}{f_T} \cdot V_{\max,X}^0 \quad (8.26)$$

$$A_X = f_{A,X} \cdot A_X^0 \quad (8.27)$$

[TODO: differential temperature sensitivity; see Smith 2013]

Protein sub-partitioning of overall uptake machinery (f_V) into affinity and transport capabilities is instantaneously optimized

$$\frac{\partial V_X^*}{\partial f_{A,X}} = 0 \quad (8.28)$$

and the resulting optimal sub-partitioning depends on nutrient availability ([Pahlow, 2005](#); [Smith et al., 2009](#))

$$f_{A,X} = \left(1 + \sqrt{A_X^0 \cdot \text{DIX} / V_{\max,X}^0} \right)^{-1} \quad (8.29)$$

In terms of the autotroph nutrient quota Q_X ($X=N, P, Si$) imbalance between nutrient uptake and growth demand leads to

$$\frac{d}{dt} Q_X = V_X - \mu \cdot Q_X \quad (8.30)$$

or, in the MAECS notation where traits/characteristics are transported as bulk biomasses Phy_X parallel to the basic phytoplankton concentration Phy_C

$$\frac{d}{dt} \text{Phy}_X = V_X \cdot \text{Phy}_C - \underset{(8.9)}{M} \cdot \text{Phy}_X \quad \text{with} \quad Q_X = \frac{\text{Phy}_X}{\text{Phy}_C} \quad (8.31)$$

[TODO: exudation]

8.2.6 Grazing

Before merging with the ecological MAECS version, we use primitive (non-biophysical) standards for describing heterotrophic activities:

$$\frac{d}{dt}Z_{\text{ooC}} = (y \cdot G - \tau_{\text{C}} - M_Z) \cdot Z_{\text{ooC}} \quad (8.32)$$

with quadratic and temperature sensitive mortality to represent top-down pressure

$$M_Z = f_T \cdot M_Z^0 \cdot Z_{\text{ooC}} \quad (8.33)$$

and a Holling-III response function for functional grazing response:

$$G = g_{\text{max}} f_T \cdot \frac{\text{Phy}_{\text{C}}^2}{K_G^2 + \text{Phy}_{\text{C}}^2} \quad (8.34)$$

[TODO: more complicated loss/leakage terms τ_{X} to ensure homeostasis and mass conservation]

8.2.7 Sinking loss

Vertical losses account for enhanced settling due to aggregate formation which occurs under high concentrations of particles and exopolymers. As a coastal model, already the basic version of MAECS resolves a benthic compartment independently from the coupled diagenesis model (see below).

The relative loss rate S is, in an idealized picture, the ratio between v_s and the mixed layer depth (MLD)

$$S = (1 + \underbrace{f_{\text{agg}}}_{(8.40)}) \cdot \frac{v_s}{\text{MLD}} \quad (8.35)$$

[TODO: add size as trait or explicit parameter; the following part comes from the size-based model version]

The velocity v_s of a sinking particle is described by Stokes' law, which for a spherical cell with diameter $\text{ESD} = e^\ell$ reads

$$v_s = \frac{g \rho'(\ell) e^{2\ell}}{18 \nu(T)} \quad (8.36)$$

The unitless excess density ρ' , the density difference between water and the suspended body, divided by water density, is known to be a function of the physiology and size of the suspended cells (Waite et al., 1997; Kjørboe et al., 1998; Miklasz and Denny, 2010). Dead cells are relatively heavy-weighted ($\rho' = \rho^\dagger$, $\rho^\dagger > 0$), but ρ^\dagger decreases in large (siliceous) phytoplankton due to an increasing fraction of vacuoles. The allometric relation of vacuolation and excess density reduction is

$$\rho^\dagger = \rho_0^\dagger e^{-\alpha_p \ell} \quad (8.37)$$

As vacuoles usually contain a higher concentration of inorganic compounds than the surrounding cytoplasm, the exponent α_p should be close to the size scaling slope reported for carbon density by Menden-Deuer and Lessard (2000).

Linear dependency between excess density and relative production

$$\rho' = (1 - f_{\text{vac}} \cdot p) \cdot \rho^\dagger \quad (8.38)$$

f_{vac} is the relative volumetric fraction that can be filled with material of reduced density (e.g., gases, lipids, or solutes, the latter listed in Boyd and Gradmann (2002)). This quantity relates to the vacuole structure (Raven and Waite, 2004) and can therefore be formulated as the relative density difference with respect to cells without vacuolation (ρ_0^\dagger in Eq.(8.37))

$$f_{\text{vac}} = \frac{\rho_0^\dagger - \rho^\dagger}{\rho_0^\dagger} = 1 - e^{-\alpha_p \ell} \quad (8.39)$$

8.2.8 Particle aggregation

[check with Richard; re-introduce into code]

Aggregation depends on stickiness and particle surface

$$f_{agg} = f_{agg}^* EP \cdot \text{Phy}_C \cdot \text{ESD}^n \quad (8.40)$$

The relative fraction of exopolymers within the DOM pool is inversely related to DOM quality, which is expressed in terms of the N:C stoichiometry. Stickiness associated with exopolymers (EP) correlates with DOM quantity and inverse quality

$$EP = \frac{\text{DOC}}{\text{DON}} \text{DOC} \quad (8.41)$$

Stickiness not only enhances coagulation efficiency in particle aggregation, it also increases the critical bottom shear stress for sediment resuspension ("biostabilization"). Resuspension of benthic material occurs when bottom shear stress exceeds that "sticky" threshold.

$$RS = RS^* \max \left\{ V_{bshear} - V_{EP}^* EP, 0 \right\} \quad (8.42)$$

so we have for material fluxes due to resuspension

$$\frac{d}{dt} C_X^{z=H} = \dots + RSC_X^{\text{ben}} \quad \text{with} \quad C_X = \text{Phy}_X, \text{Det}_X \quad (8.43)$$

8.2.9 Other stuff

Photoinhibition by depletion of D1-protein

$$u = \frac{q_N}{q_N + u^* \text{Chl:C}^2} \quad (8.44)$$

Exudation reflects imbalance between C uptake and assimilation

$$E = e^* P_{\max}^* f_T \text{ LH} \quad (8.45)$$

8.3 Adaptive trait regulation and differential trade-off

Adaptive trait dynamics in its general form has been proposed as an optimality-seeking principle guiding transient adaptive regulation phenomena on very different levels of description, from organ physiology to population ecology (Wirtz, 2000, 2003; Smith et al., 2011). This principle/equation is applied to all physiological traits in MAECS, in particular to those that primarily control nutrient uptake (f_R and θ through f_V , and activity $a_{V,X}$). C-growth as goal function needs to be extended by "hidden" or indirect effects through a differential link between nutrient uptake V_X and quota Q_X for all macro-nutrients:

$$\frac{df_{m,X}}{dt} = \delta_m \left(\frac{\partial \mu}{\partial f_{m,X}} + \sum_x \underbrace{\frac{\partial \mu}{\partial Q_X}}_{(8.56)} \underbrace{\frac{dQ_X}{dV_X}}_{(8.57)} \bigg|_{\text{tot}} \frac{\partial V_X}{\partial f_{m,X}} \right) \quad \text{with} \quad f_{m,X} = f_R, \theta, a_{V,X} \quad X = \text{N, P, Si, } \dots \quad (8.46)$$

[TODO: reduce text or spread to individual euqations]

Diverging from most standard ecosystem models, however, MAECS assumes intricate interdependencies between independent C, N, or P assimilation functions while avoiding prescribed stoichiometric control settings such as maximal N:C or P:C ratios. For doing so, it implies optimality criteria already in the description of basic uptake formulations (lower part of Table ??). These criteria control shifts between high affinity and fast transport nutrient uptake (see above Eq.(8.29)) and in the enzymatic down-regulation of all nutrient uptake activities. Adaptive control in activity secures phytoplankton cells from non-beneficial intracellular accumulation of nutrients, but in the

model also requires to formulate an extended optimality principle by which C costs of nutrient uptake have to be balanced with corresponding C benefits arising from associated quota changes (Eq.(??)). For both regulations (uptake site/transport partitioning, activity), steady-state solutions or approximations are calculated since physiological uptake regulations proceed at very high speed.

For nitrogen (N), the differential effect of increasing nutrient uptake rate on the quota derives from functional variation applied to the quota uptake equation Eq.(8.30):

$$\delta V_X + \frac{\partial V_X}{\partial Q_X} \delta Q_X - \mu \cdot \delta Q_X - Q_X \cdot \frac{\partial \mu}{\partial Q_X} \delta Q_X - Q_X \cdot \frac{\partial \mu}{\partial V_X} \delta V_X = 0 \quad (8.47)$$

or

$$\frac{dQ_X}{dV_X} = (1 + \zeta_X Q_X) \cdot \left(\mu + Q_X \frac{\partial \mu}{\partial Q_X} - \frac{\partial V_X}{\partial Q_X} \right)^{-1} \quad (8.48)$$

where the derivative of uptake rate on quota may only become non-zero (1) for nitrogen (N) and (2) if C-partitioning to chlorophyll (f_θ in Eq.(8.21)) is hardwired to the N-quota:

$$\frac{\partial V_N}{\partial Q_N} = \frac{V_N}{f_V} \frac{\partial f_V}{\partial Q_N} = -\frac{\sigma f_R \theta}{f_V \theta_C \Delta Q_N} V_N = -\sigma' V_N \quad (8.49)$$

8.3.1 Growth derivatives under co-limitation

For the N-turnover and regulation, the differential dependency of V_N on Q_N also enters the marginal increase in primary production when raising intracellular quota; an analytical derivation of the photosynthesis rate P in Eq.(8.10) with respect to each co-limiting quota Q_X reads

$$\begin{aligned} \frac{\partial \mu}{\partial Q_X} &= \frac{P}{LF} \frac{\partial LF}{\partial Q_X} \frac{\partial Q_X}{\partial Q_X} - \zeta \frac{\partial V_N}{\partial Q_X} \\ &= d_X \frac{\mu + \zeta V_N}{\Delta Q_X} + \sigma' \zeta V_N \quad \text{with } d_X = LF^{-1} \frac{\partial LF}{\partial Q_X} \\ &= \left(d_X \frac{1 + \zeta Q_N}{\Delta Q_X} + \sigma' \zeta Q_N \right) \cdot \mu \\ &= \quad \quad \quad d_{QX} \quad \cdot \mu \end{aligned} \quad (8.50)$$

where we assumed a balanced growth relation between growth and uptake ($\mu Q_N = V_N$), used the relation $P = \mu + \zeta V_N$. In light of Eq.(8.49) we have $\sigma'_N = \sigma'$ and $\sigma'_X = 0$ for other elements that lack direct influence on N-uptake rate.

The derivation result d_X of the (recursive) co-limitation factor $LF \equiv q'_1$ may contain a number of product terms, depending on where in the scheme the limiting effect of Q_X is calculated. Consider the series of limitation factors $q_1, q_2, \dots, q_{N_{\text{out}}}$ (e.g., for q_N, q_P, q_{Si}) we start from the first element where the recursive scheme Eq.(8.18) has been invoked only once. For example, if MAECS just resolves its basic element N,

$$d_N = \frac{1}{q'_N} \quad \text{for } N_{\text{out}} = 1 \quad (8.51)$$

or N and P,

$$d_N = \frac{1}{q'_N} \frac{\partial q'_N}{\partial q_N} \quad d_P = \frac{1}{q'_N} \frac{\partial q'_N}{\partial q'_P} \frac{\partial q'_P}{\partial q_P} \quad \text{for } N_{\text{out}} = 2 \quad (8.52)$$

Note that $\partial q'_P / \partial q_P$ is one because P makes the last element in the list, so that the metabolic effect of Q_P as quantified by q'_P exclusively depends on the availability of Q_P as quantified by q_P . The general form for an arbitrary N_{out} continues the sequential differentiation from the first element (here usually N) to the element X under consideration. So starting from q'_1 we calculate the use efficiency of element X again using the chain-rule:

$$d_X = \frac{1}{q'_1} \frac{\partial q'_1}{\partial q'_2} \frac{\partial q'_2}{\partial q'_3} \dots \frac{\partial q'_X}{\partial q_X} \quad (8.53)$$

Note that the last term in the product of differentials ($\partial q'_X / \partial q_X$) is either one if X is the last element in the sequence, and otherwise given by Eq.(8.54). The differentials in Eqs.(8.51)–(8.53) characterize the recursive effect of metabolic efficiencies of q'_X formulated in Eq.(8.18)

$$\frac{\partial q'_X}{\partial q_X} = q'_X \cdot \left(q_X^{-1} - \frac{\partial g_h}{\partial x} \frac{q'_{X+1}}{g_h q_X^2} + \frac{h q'_{X+1}}{c_{hq}} \right) \quad (8.54)$$

or, if we differentiate with respect to the second efficiency:

$$\frac{\partial q'_X}{\partial q'_{X+1}} = q'_X \cdot \left(\frac{\partial g_h}{\partial x} \frac{1}{g_h q_X} + \frac{h q_X}{c_{hq}} \right) \quad (8.55)$$

where the coefficient c_{hq} given in Eq.(8.14) is written without nutrient specific indices. Step-wise derivation of the queuing function $(x - x_h) / (1 - x_h)$ with $x_h = x^{1+h^{-1}}$ in Eq.(8.16) yields

$$\begin{aligned} \frac{\partial g_h}{\partial x} &= \frac{(1 - (1 + h^{-1})x_h/x) \cdot (1 - x_h) + (x - x_h) \cdot (1 + h^{-1})x_h/x}{(1 - x_h)^2} \\ &= \frac{(xh - (h+1)x_h) \cdot (1 - x_h) + (x - x_h) \cdot (h+1)x_h}{xh \cdot (1 - x_h)^2} \\ &= \frac{xh - (h+1)x_h - \cancel{xhx_h} + \cancel{(h+1)x_h^2} + x \cdot \cancel{(h+1)x_h} - \cancel{(h+1)x_h^2}}{xh \cdot (1 - x_h)^2} \\ &= \frac{xh + (x - 1 - h) \cdot x_h}{xh \cdot (1 - x_h)^2} \end{aligned} \quad (8.56)$$

[TODO: numerical approximation to avoid problems at $x = 1$]

8.3.2 Quota-uptake feed-back

Eq.(8.48) provides a first estimate for the differential trade-off required for a fully coherent application of the optimality principle to physiological regulation. However, a marginal change in quota after a differential change in uptake rate may propagate back to the uptake rate, if the latter directly depends on Q_N . This direct, differential feed-back between changes in Q_N and V_N reads

$$\left. \frac{dQ_N}{dV_N} \right|_{\text{tot}} = \frac{dQ_N}{dV_N} \cdot \left(1 + \underbrace{\frac{dQ_N}{dV_N} \frac{\partial V_N}{\partial Q_N}}_{(8.58)} \right) \quad (8.57)$$

$$\begin{aligned} \frac{dQ_N}{dV_N} \frac{\partial V_N}{\partial Q_N} &= -(1 + \zeta Q_N) \cdot \left(\mu + Q_N \frac{\partial \mu}{\partial Q_N} - \frac{\partial V_N}{\partial Q_N} \right)^{-1} \sigma' V_N \\ &= -(1 + \zeta Q_N) \cdot \left(1 + Q_N \left[d_N \frac{1 + \zeta Q_N}{Q_N - Q_{N0}} + \sigma' \zeta Q_N \right] + \sigma' Q_N \right)^{-1} \sigma' Q_N \\ &= -(1 + \zeta Q_N) \cdot \left((Q_N)^{-1} + d_N \frac{1 + \zeta Q_N}{Q_N - Q_{N0}} + \sigma' \cdot (1 + \zeta Q_N) \right)^{-1} \sigma' \\ &= -\sigma' \cdot \underbrace{\left((Q_N (1 + \zeta Q_N))^{-1} + d_N (Q_N - Q_{N0})^{-1} + \sigma' \right)}_{e_N}^{-1} \\ &= -\frac{\sigma'}{e_N + \sigma'} \end{aligned} \quad (8.58)$$

In all other cases apart of nitrogen, the uptake dependency on the quota vanishes:

$$\frac{\partial V_X}{\partial Q_X} = 0 \quad \text{and} \quad \left. \frac{dQ_X}{dV_X} \right|_{\text{tot}} = \frac{dQ_X}{dV_X} \quad \text{with} \quad X = P, Si \quad (8.59)$$

The product of the quota-uptake differential (without feed-back) and the growth-quota differential in Eq.(8.46) combines Eq.(8.48) and Eq.(8.50) and again assumes $V_X = Q_X \mu$:

$$\begin{aligned}
 \frac{dQ_X}{dV_X} \frac{\partial \mu}{\partial Q_X} &= (1 + \zeta_X Q_X) \cdot \left(\mu + Q_X \frac{\partial \mu}{\partial Q_X} - \frac{\partial V_X}{\partial Q_X} \right)^{-1} \cdot d_{QX} \cdot \mu \\
 &= (1 + \zeta_X Q_X) \cdot \left(1 + Q_X d_{QX} + \sigma'_X V_X \mu^{-1} \right)^{-1} \cdot d_{QX} \\
 &= \frac{(1 + \zeta_X Q_X) \cdot d_{QX}}{1 + Q_X \cdot (d_{QX} + \sigma'_X)} \quad (8.60)
 \end{aligned}$$

8.3.3 Uptake activity regulation

In the current version of MAECS, regulation of all uptake activity traits a_X is supposed to be very fast compared to the simulated dynamics and therefore not integrated in time according to Eq.(8.46), but assumed to be in steady-state. If the marginal benefit of uptake $d\mu/da_X$ is negative, activity is ceased; at positive benefit, a_X approaches one. While at neutral growth effect $d\mu/da_X \approx 0$, the activity should for numerical reasons smoothly decrease to a small number. This behavior is emulated by the non-linear function

$$a_X = \left(1 + e^{\pi - \Delta \mu / da_X} \right)^{-1} \quad (8.61)$$

For optimization in N-uptake activity a_N based on its the marginal C gain the extended optimality principle integrates Eq.(8.46) and Eqs.(8.50)–(8.60):

$$\begin{aligned}
 \frac{d\mu}{da_X} &= \frac{\partial \mu}{\partial a_X} + \frac{\partial \mu}{\partial Q_X} \frac{dQ_X}{dV_X} \Big|_{\text{tot}} \frac{\partial V_X}{\partial a_X} \\
 &= -\zeta_X \frac{V_X}{a_X} + \frac{\partial \mu}{\partial Q_X} \frac{dQ_X}{dV_X} \frac{e_N}{e_N + \sigma'_X} \frac{V_X}{a_X} \\
 &= \left(-\zeta_X + \frac{(1 + \zeta_X Q_X) \cdot d_{QX}}{1 + Q_X \cdot (d_{QX} + \sigma'_X)} \frac{e_N}{e_N + \sigma'_X} \right) \cdot \frac{V_X}{a_X} \quad (8.62)
 \end{aligned}$$

where, again, $\sigma'_N = \sigma'$ and $\sigma'_X = 0$ for other elements that lack direct influence on uptake.

8.3.4 Costs in P and Si uptake

For a first estimation of the C-costs of P- and Si-uptake (with units mol-C/mol-X) we link the latter to N-assimilation. This means that energetic costs of P- and Si-assimilation are not accounted for as additional terms but assumed to be already included in protein synthesis that are chararized by $\zeta \equiv \zeta_N$ (with units mol-C/mol-N). For the P-link, we use the N-stoichiometry in RNA (N:P \approx 3.8:1) and phospholipids (N:P \approx 0.8:1 mol-N/mol-P)

$$\frac{\partial \mu}{\partial V_X} = \frac{\partial \mu}{\partial V_N} \frac{\partial V_N}{\partial V_X} = -\zeta_N \cdot \frac{Q_N^{0*}}{Q_X^{0*}} = -\zeta_X \quad (8.63)$$

with Eq.(8.14) Eq.(8.18)

$$\zeta_P = \left[(1 - f_{\text{Lip}}) 3.8 + f_{\text{Lip}} 0.8 \right] \cdot \zeta_N \quad \text{and} \quad \zeta_{\text{Si}} = 0 \quad (8.64)$$

[TODO: check and simplify]

[TODO: include proteins/membranes (N:P \gg 16:1) under low growth conditions]

8.3.5 Photoacclimation and transport

MAECS resolves transient photoacclimation as adaptive dynamics in allocation traits (Eq.(8.46)). The optimality principle extended by the differential quota-based trade-off Eq.(8.60) seeks to find an allocation key between nutrient uptake, LHC and light-independent processes (Rubisco) that maximizes relative C-uptake rate μ . The optimality condition includes marginal growth benefits of all nutrients (see Eq.(8.46)):

$$\frac{d}{dt} f_R = \delta_R \left(\frac{\partial \mu}{\partial f_R} + \sum_x \frac{\partial \mu}{\partial Q_X} \frac{dQ_X}{dV_X} \Big|_{\text{tot}} \frac{\partial V_X}{\partial f_R} \right) \quad (8.65)$$

(8.56) (8.57) (8.68)

and similar for the chloroplast CHL:C ratio θ :

$$\frac{d}{dt} \theta = \delta_\theta \left(\frac{\partial \mu}{\partial \theta} + \sum_x \frac{\partial \mu}{\partial Q_X} \frac{dQ_X}{dV_X} \Big|_{\text{tot}} \frac{\partial V_X}{\partial \theta} \right) \quad (8.66)$$

(8.50) (8.57) (8.69)

The differential growth loss by increasing allocation to photosynthesis apparati down-sizes the nutrient uptake machinery. All uptake and indirect derivatives of the photoacclimation traits that induce these differential costs had been already introduced above.

Flexibilities in chloroplast CHL:C ratio and in C-allocation to Rubisco are given following [Wirtz and Eckhardt \(1996\)](#); [Wirtz \(2000\)](#)

$$\delta_\theta = \delta_\theta^* \cdot \theta \cdot (\theta_C - \theta) \quad \delta_R = \delta_R^* \cdot f_R \cdot (1 - f_R) \quad (8.67)$$

Partial derivatives of photosynthesis rates with respect to θ and f_R (see Eq.(8.20) and Eq.(8.22)):

$$\begin{aligned} \frac{\partial \mu}{\partial f_R} &= \frac{P}{f_R} - \zeta \frac{\partial V_N}{\partial f_R} \\ &= \frac{P}{f_R} + \zeta \cdot \left(1 + \frac{q_N^\sigma \theta}{\theta_C} \right) \cdot a_{V,N} V_X^* \end{aligned} \quad (8.68)$$

(8.61)

$$\begin{aligned} \frac{\partial \mu}{\partial \theta} &= \frac{P}{LH} \frac{\partial LH}{\partial \theta} - \zeta \frac{\partial V_N}{\partial \theta} \\ &= \frac{P}{LH} \frac{\alpha \text{PAR}}{P_{\max}} (1 - LH) - \zeta \frac{q_N^\sigma f_R}{\theta_C} \cdot a_{V,N} V_X^* \end{aligned} \quad (8.69)$$

For transporting photoacclimation traits in 1D-3D, MAECS integrates them as bulk variables by employing a "carrier" biomass variable (usually Phy_C). With Eq.(8.21) we have for the bulk chlorophylla concentration

$$\text{Chl} = f_\theta \theta_C \text{Phy}_C = f_R q_N^\sigma \theta \text{Phy}_C \quad (8.70)$$

and bulk Rubisco concentrations

$$\text{Rub} = f_R \text{Phy}_C \quad (8.71)$$

[TODO: write down full equations]

8.4 Acknowledgements

We thank The work was supported by the Helmholtz society via the program PACES.

Bibliography

- C. Boyd and D. Gradmann. Impact of osmolytes on buoyancy of marine phytoplankton. 141(4):605–618, 2002. [33](#)
- A. D. Friend. Use of a model of photosynthesis and leaf microenvironment to predict optimal stomatal conductance and leaf nitrogen partitioning. 14:895–905, 1991. [31](#)
- T. Kjørboe, P. Tiselius, B. Mitchell-Innes, J.L.S. Hansen, A.W. Visser, and X. Mari. Intensive aggregate formation with low vertical flux during an upwelling-induced diatom bloom. 43:104–116, 1998. [33](#)
- S. Menden-Deuer and E. J. Lessard. Carbon to volume relationships for dinoflagellates, diatoms, and other protist plankton. 45(3):569–579, 2000. [33](#)
- K.A. Miklasz and M.W. Denny. Diatom sinking speeds: improved predictions and insight from a modified Stokes' law. 55(6):2513–2525, 2010. [33](#)
- F. M. M. Morel. Kinetics of nutrient uptake and growth in phytoplankton. 23(2):137–150, 1987. [30](#)
- M. Pahlow. Linking chlorophyll-nutrient dynamics to the Redfield N: C ratio with a model of optimal phytoplankton growth. 287:33–43, 2005. [30](#), [32](#)
- J. A. Raven and A. M. Waite. The evolution of silicification in diatoms: inescapable sinking and sinking as escape? 162:45–61, 2004. [33](#)
- JA Raven. Dark respiration. In *Energetics and transport in aquatic plants*, volume 4 of *MBL Lectures*, pages 253–317. Alan R. Liss; Inc., New York, 1984. [30](#)
- S. L. Smith, Y. Yamanaka, M. Pahlow, and A. Oschlies. Optimal uptake kinetics: physiological acclimation explains the pattern of nitrate uptake by phytoplankton in the ocean. 384:1–12, 2009. [32](#)
- S. L. Smith, M. Pahlow, A. Merico, and K. W. Wirtz. Optimality as a unifying concept for planktonic organisms and their ecology. 56:2080–2094, 2011. [34](#)
- A. Waite, A. Fisher, P.A. Thompson, and P.J. Harrison. Sinking rate verses cell volume relationships illuminate sinking rate control mechanisms in marine diatoms. 157:97–108, 1997. [33](#)
- K. W. Wirtz. Simulating the dynamics of leaf physiology and morphology with an extended optimality approach. 86:753–764, 2000. [34](#), [38](#)
- K. W. Wirtz. Adaptive significance of C partitioning and SLA regulation in *Betula pendula*. 23(3):181–190, 2 2003. [34](#)
- K. W. Wirtz. Non-uniform scaling in phytoplankton growth rate due to intracellular light and CO₂ decline. 33:1325–1341, 2011. [29](#)
- K. W. Wirtz. Intermittency in processing explains the diversity and shape of functional grazing responses. 169:879–894, 2012. [30](#)
- K. W. Wirtz. Mechanistic origins of variability in phytoplankton dynamics. Part I: Niche formation revealed by a size-based model. DOI 10.1007/s00227-012-2163-7, 2013. doi: 10.1007/s00227-012-2163-7. [29](#)
- K. W. Wirtz and B. Eckhardt. Effective variables in ecosystem models with an application to phytoplankton succession. 92:33–53, 1996. [38](#)
- K. W. Wirtz and M. Pahlow. Dynamic CHL and N–C regulation in algae optimizes instantaneous growth rate. 402:81–96, 2010. [4](#), [31](#)

Index

- calc_internal_states
 - maecs_functions, 14
- calc_sensitivities
 - maecs_functions, 15
- fabm_hzg_maecs, 9
 - get_light_extinction, 11
 - initialize, 10
 - maecs_do, 11
 - maecs_get_vertical_movement, 13
 - maecs_init_stoichvars, 11
- fabm_hzg_maecs::type_hzg_maecs, 20
- foptupt
 - maecs_functions, 15
- get_light_extinction
 - fabm_hzg_maecs, 11
- grazing
 - maecs_grazing, 18
- grazing_losses
 - maecs_grazing, 18
- initialize
 - fabm_hzg_maecs, 10
- maecs_do
 - fabm_hzg_maecs, 11
- maecs_functions, 13
 - calc_internal_states, 14
 - calc_sensitivities, 15
 - foptupt, 15
 - min_mass, 16
 - queuederiv, 16
 - queuefunc, 15
 - queuefunc1, 16
 - sinking, 16
 - smooth_small, 17
 - uptflex, 15
- maecs_functions.F90, 25
- maecs_get_vertical_movement
 - fabm_hzg_maecs, 13
- maecs_grazing, 17
 - grazing, 18
 - grazing_losses, 18
- maecs_grazing.F90, 25
- maecs_init_stoichvars
 - fabm_hzg_maecs, 11
- maecs_maecsdo_combined.F90, 26
- maecs_primprod, 18
 - photosynthesis, 18
- maecs_primprod.F90, 26
- maecs_types, 19
- maecs_types.F90, 26
- maecs_types::stoich_pointer, 20
- maecs_types::type_maecs_allocation_fractions, 21
- maecs_types::type_maecs_base_model, 21
- maecs_types::type_maecs_basic_traits, 21
- maecs_types::type_maecs_env, 21
- maecs_types::type_maecs_life, 22
- maecs_types::type_maecs_nutindex, 22
- maecs_types::type_maecs_om, 22
- maecs_types::type_maecs_phy, 23
- maecs_types::type_maecs_rhs, 23
- maecs_types::type_maecs_sensitivities, 23
- maecs_types::type_maecs_switch, 23
- maecs_types::type_maecs_traitdyn, 24
- maecs_types::type_maecs_zoo, 24
- min_mass
 - maecs_functions, 16
- photosynthesis
 - maecs_primprod, 18
- queuederiv
 - maecs_functions, 16
- queuefunc
 - maecs_functions, 15
- queuefunc1
 - maecs_functions, 16
- sinking
 - maecs_functions, 16
- smooth_small
 - maecs_functions, 17
- uptflex
 - maecs_functions, 15