



Hochschule
Zittau/Görlitz
UNIVERSITY OF APPLIED SCIENCES

Faculty of
MECHANICAL ENGINEERING

Department of
TECHNICAL THERMODYNAMICS

Property Library for Seawater Calculated from the IAPWS Industrial Formulation 2013

**FluidMAT
with LibSeaWa
for Mathcad®**

Prof. Hans-Joachim Kretzschmar

Dr. Ines Stoecker

Thomas Gubsch

Matthias Kunick

Tobias Goepfert

Property Library for Seawater Calculated the IAPWS Industrial Formulation 2013

LibSeaWa FluidMAT for Mathcad®

Contents

- 0 Package Contents
- 1 Property Functions
- 2 Application of FluidMAT in Mathcad®
 - 2.1 Installing FluidMAT LibSeaWa
 - 2.2 Licensing the LibSeaWa Property Library
 - 2.3 Example: Calculation of $h = f(p, t, \xi)$
 - 2.4 Removing FluidMAT
- 3 Program Documentation
- 4 Property Libraries for Calculating Heat Cycles, Boilers, Turbines, and Refrigerators
- 5 References
- 6 Satisfied Customers

© Zittau/Goerlitz University of Applied Sciences
Faculty of Mechanical Engineering
Department of Technical Thermodynamics
Professor Hans-Joachim Kretzschmar
Dr. Ines Stoecker
Phone: +49 3583-61-1846 or -1881
Fax: +49 3583-61-1846
E-mail: hj.kretzschmar@hs-zigr.de
Internet: www.thermodynamics-zittau.de

0 Package Contents

CD "FluidMAT LibSeaWa for Mathcad®" for calculating the properties of seawater
including the following files:

FluidMAT_LibSeaWa_Setup.exe	Self-extracting installation program
FluidMAT_LibSeaWa_Docu.pdf	User's Guide
LibSeaWa.dll	DLL including functions of the LibSeaWa library
MAT_LibSeaWa_DE.xml	Function registration in the dialog window "Insert Function" for German Mathcad 12 or higher version
MAT_LibSeaWa_EN.xml	Function registration in the dialog window "Insert Function" for English Mathcad 12 or higher version
MAT_LibSeaWa.xml	Function registration in the dialog window "Insert Function" for Mathcad version 11 or lower version

1. Property Functions

1.1 Functions

Functional Dependence	Function Name in Excel®	Call as Fortran Program	Property or Function	Unit of the Result	Reference	Page
$a = f(p, t, \xi)$	a_ptXI_SeaWa	= A_PT XI_SEAWA(P,T,XI)	Thermal diffusivity	m ² /s	[1], [2]	3/2
$a_l = f(t, \xi)$	al_tXI_SeaWa	= AL_T XI_SEAWA(T,XI)	Thermal diffusivity of subcooled liquid	m ² /s	[1], [2]	3/3
$a_{sl} = f(p_s, t_s, \xi_{sl})$	asl_pstsXisl_SeaWa	= ASL_PST SXISL_SEAWA(PS,TS,XISL)	Thermal diffusivity of saturated liquid	m ² /s	[1], [2]	3/4
$a_{sv} = f(p_s, t_s, \xi_{sl})$	asv_pstsXisl_SeaWa	= ASV_PST SXISL_SEAWA(PS,TS,XISL)	Thermal diffusivity of saturated vapor	m ² /s	[1], [2]	3/5
$\alpha_l = f(p, t, \xi)$	al pha_l_ptXi_SeaWa	= AL PHAL_PT XI_SEAWA(P,T,XI)	Thermal expansion coefficient of subcooled liquid	1/K	[1]	3/6
$\alpha_{sl} = f(p_s, t_s, \xi_{sl})$	al pha_sl_pstsXisl_SeaWa	= AL PHASL_PST SXISL_SEAWA(PS,TS,XISL)	Thermal expansion coefficient of saturated liquid	1/K	[1]	3/7
$\beta_l = f(p, t, \xi)$	be ta_l_ptXi_SeaWa	= BE TAL_PT XI_SEAWA(P,T,XI)	Haline contraction coefficient of subcooled liquid	kg/kg	[1]	3/8
$\beta_{sl} = f(p, t, \xi_{sl})$	be ta_sl_pstsXisl_SeaWa	= BE TASL_PST SXISL_SEAWA(PS,TS,XISL)	Haline contraction coefficient of saturated liquid	kg/kg	[1]	3/9
$\beta_{lsl} = f(p, t, \xi)$	be ta_lsl_ptXi_SeaWa	= BE TAISL_PT XI_SEAWA(P,T,XI)	Isentropic temperature-pressure coefficient of subcooled liquid	K/kPa	[1]	3/10
$\beta_{lssl} = f(p, t, \xi_{sl})$	be ta_lssl_pstsXisl_SeaWa	= BE TAISSL_PST SXISL_SEAWA(PS,TS,XISL)	Isentropic temperature-pressure coefficient of saturated liquid	K/kPa	[1]	3/11
$c_p = f(p, t, \xi)$	cp_ptXI_SeaWa	= CP_PT XI_SEAWA(P,T,XI)	Specific isobaric heat capacity	kJ/(kg·K)	[1], [2]	3/12
$c_{p,l} = f(p, t, \xi)$	cpl_ptXI_SeaWa	= CPL_PT XI_SEAWA(P,T,XI)	Specific isobaric heat capacity of subcooled liquid	kJ/(kg·K)	[1], [2]	3/13
$c_{p,sl} = f(p_s, t_s, \xi_{sl})$	cpsl_pstsXisl_SeaWa	= CPSL_PST SXISL_SEAWA(PS,TS,XISL)	Specific isobaric heat capacity of saturated liquid	kJ/(kg·K)	[1], [2]	3/14

Functional Dependence	Function Name in Excel®	Call as Fortran Program	Property or Function	Unit of the Result	Reference	Page
$c_{p_{sv}} = f(p_s, t_s, \xi_{sl})$	cpsv_pstsXisl_SeaWa	= CPSV_PSTSXISL_SEAWA(P,S,XISL)	Specific isobaric heat capacity of saturated vapor	kJ/(kg·K)	[1], [2]	3/15
$\eta = f(p, t, \xi)$	eta_ptXI_SeaWa	= ETA_PTXI_SEAWA(P,T,XI)	Dynamic viscosity	Pa·s	[1], [2]	3/16
$\eta_l = f(t, \xi)$	etal_tXI_SeaWa	= ETAL_TXI_SEAWA(T,XI)	Dynamic viscosity of subcooled liquid	Pa·s	[1], [2]	3/17
$\eta_{sl} = f(p_s, t_s, \xi_{sl})$	etasl_pstsXisl_SeaWa	= ETASL_PSTSXISL_SEAWA(P,S,XISL)	Dynamic viscosity of saturated liquid	Pa·s	[1], [2]	3/18
$\eta_{sv} = f(p_s, t_s, \xi_{sl})$	etasv_pstsXisl_SeaWa	= ETASV_PSTSXISL_SEAWA(P,S,XISL)	Dynamic viscosity of saturated vapor	Pa·s	[1], [2]	3/19
$f_l = f(p, t, \xi)$	fl_ptXI_SeaWa	= FL_PTXI_SEAWA(P,T,XI)	Specific Helmholtz energy of subcooled liquid	kJ/kg	[1]	3/20
$f_{sl} = f(p_s, t_s, \xi_{sl})$	fsl_pstsXisl_SeaWa	= FSL_PSTSXISL_SEAWA(P,S,XISL)	Specific Helmholtz energy of saturated liquid	kJ/kg	[1]	3/21
$\phi = f(p, t, \xi)$	phil_ptXI_SeaWa	= PHIL_PTXI_SEAWA(P,T,XI)	Osmotic coefficient of subcooled liquid	[-]	[1]	3/22
$\phi_{sl} = f(p_s, t_s, \xi_{sl})$	phisl_pstsXisl_SeaWa	= PHISL_PSTSXISL_SEAWA(P,S,XISL)	Osmotic coefficient of saturated liquid	[-]	[1]	3/23
$h = f(p, t, \xi)$	h_ptXI_SeaWa	= H_PTXI_SEAWA(P,T,XI)	Specific enthalpy	kJ/kg	[1], [2]	3/24
$h_l = f(p, t, \xi)$	hl_ptXI_SeaWa	= HL_PTXI_SEAWA(P,T,XI)	Specific enthalpy of subcooled liquid	kJ/kg	[1], [2]	3/25
$h_{sl} = f(p_s, t_s, \xi_{sl})$	hsl_pstsXisl_SeaWa	= HSL_PSTSXISL_SEAWA(P,S,XISL)	Specific enthalpy of saturated liquid	kJ/kg	[1], [2]	3/26
$h_{sv} = f(p_s, t_s, \xi_{sl})$	hsv_pstsXisl_SeaWa	= HSV_PSTSXISL_SEAWA(P,S,XISL)	Specific enthalpy of saturated steam	kJ/kg	[1], [2]	3/27
$\kappa = f(p, t, \xi)$	kappa_ptXI_SeaWa	= KAPPA_PTXI_SEAWA(P,T,XI)	Isentropic exponent	[-]	[1]	3/28
$\kappa_l = f(p, t, \xi)$	kappal_ptXI_SeaWa	= KAPPAL_PTXI_SEAWA(P,T,XI)	Isentropic exponent of subcooled liquid	[-]	[1]	3/29
$\kappa_{sl} = f(p_s, t_s, \xi_{sl})$	kappasl_pstsXisl_SeaWa	= KAPPASL_PSTSXISL_SEAWA(P,S,XISL)	Isentropic exponent of saturated liquid	[-]	[1]	3/30

Functional Dependence	Function Name in Excel®	Call as Fortran Program	Property or Function	Unit of the Result	Reference	Page
$\kappa_{sv} = f(p_s, t_s, \xi_{sl})$	kappasv_pstsXisl_SeaWa	= KAPPASV_PSTSXISL_SEAWA(P,TS,XISL)	Isentropic exponent of saturated vapor	[-]	[1]	3/31
$\kappa_{T_l} = f(p, t, \xi)$	kappaTI_ptXI_SeaWa	= KAPPATL_PTXI_SEAWA(P,T,XI)	Isothermal compressibility of subcooled liquid	1/kPa	[1]	3/32
$\kappa_{T_sl} = f(p_s, t_s, \xi_{sl})$	kappaTsl_pstsXisl_SeaWa	= KAPPATSL_PSTSXISL_SEAWA(P,TS,XISL)	Isothermal compressibility of saturated liquid	1/kPa	[1]	3/33
$\kappa_{is_l} = f(p, t, \xi)$	kappaisl_ptXI_SeaWa	= KAPPAISL_PTXI_SEAWA(P,T,XI)	Isentropic compressibility of subcooled liquid	1/kPa	[1]	3/34
$\kappa_{is_sl} = f(p_s, t_s, \xi_{sl})$	kappaissl_pstsXisl_SeaWa	= KAPPAISSL_PSTSXISL_SEAWA(P,TS,XISL)	Isentropic compressibility of saturated liquid	1/kPa	[1]	3/35
$\lambda = f(p, t, \xi)$	lambda_ptXI_SeaWa	= LAMBDA_PTXI_SEAWA(P,T,XI)	Thermal conductivity	W/(m*K)	[3], [4], [15]	3/36
$\lambda_l = f(t, \xi)$	lambdal_tXI_SeaWa	= LAMBDA_L_PTXI_SEAWA(T,XI)	Thermal conductivity of subcooled liquid	W/(m*K)	[3], [4], [15]	3/37
$\lambda_{sl} = f(p_s, t_s, \xi_{sl})$	lambdasl_pstsXisl_SeaWa	= LAMBDA_SL_PSTSXISL_SEAWA(P,TS,XISL)	Thermal conductivity of saturated liquid	W/(m*K)	[3], [4], [15]	3/38
$\lambda_{sv} = f(p_s, t_s, \xi_{sl})$	lambdasv_pstsXisl_SeaWa	= LAMBDA_SV_PSTSXISL_SEAWA(P,TS,XISL)	Thermal conductivity of saturated vapor	W/(m*K)	[3], [4], [15]	3/39
$\mu_l = f(p, t, \xi)$	myl_ptXI_SeaWa	= MYL_PTXI_SEAWA(P,T,XI)	Relative chem. potential of subcooled liquid	kJ/kg	[1]	3/40
$\mu_{sl} = f(p_s, t_s, \xi_{sl})$	mysl_pstsXisl_SeaWa	= MYSL_PSTSXISL_SEAWA(P,TS,XISL)	Relative chem. potential of saturated liquid	kJ/kg	[1]	3/41
$\mu_{w_l} = f(p, t, \xi)$	mywl_ptXI_SeaWa	= MYWL_PTXI_SEAWA(P,T,XI)	Relative chem. potential of H ₂ O of subcooled liquid	kJ/kg	[1]	3/42
$\mu_{w_sl} = f(p_s, t_s, \xi_{sl})$	mywsl_pstsXisl_SeaWa	= MYWSL_PSTSXISL_SEAWA(P,TS,XISL)	Relative chem. potential of H ₂ O of saturated liquid	kJ/kg	[1]	3/43
$\mu_{salt_l} = f(p, t, \xi)$	mySaltl_ptXI_SeaWa	= MYSALT_L_PTXI_SEAWA(P,T,XI)	Relative chem. potential of sea salt of subcooled liquid	kJ/kg	[1]	3/44
$\mu_{salt_sl} = f(p_s, t_s, \xi_{sl})$	mySaltsl_pstsXisl_SeaWa	= MYSALT_SL_PSTSXISL_SEAWA(P,TS,XISL)	Relative chem. potential of sea salt of saturated liquid	kJ/kg	[1]	3/45

Functional Dependence	Function Name in Excel®	Call as Fortran Program	Property or Function	Unit of the Result	Reference	Page
$\nu = f(p, t, \xi)$	ny_ptXI_SeaWa	= NY_PT XI_SEAWA(P,T,XI)	Kinematic viscosity	m ² /s	[1], [2]	3/46
$\nu_l = f(t, \xi)$	nyl_tXI_SeaWa	= NYL_TXI_SEAWA(T,XI)	Kinematic viscosity of subcooled liquid	m ² /s	[1], [2]	3/47
$\nu_{sl} = f(p_s, t_s, \xi_{sl})$	nysl_pstsXisl_SeaWa	= NYSL_PST SXISL_SEAWA(PS,TS,XISL)	Kinematic viscosity of saturated liquid	m ² /s	[1], [2]	3/48
$\nu_{sv} = f(p_s, t_s, \xi_{sl})$	nysv_pstsXisl_SeaWa	= NYSV_PST SXISL_SEAWA(PS,TS,XISL)	Kinematic viscosity of saturated vapor	m ² /s	[1], [2]	3/49
$p_s = f(t_s, \xi_{sl})$	ps_tsXisl_SeaWa	= PS_TSXISL_SEAWA(TS,XISL)	Boiling pressure	bar	[1] ,[2] ,[4]	3/50
$p_{mel} = f(t, \xi)$	pmel_tXi_SeaWa	= PMEL_TXI_SEAWA(T,XI)	Freezing pressure	bar	[1] ,[4], [5]	3/51
$p_{tr} = f(\xi)$	ptr_Xi_SeaWa	= PTR_XI_SEAWA(T,XI)	Triple point pressure	bar	[1] ,[4], [5]	3/52
$Pr = f(p, t, \xi)$	Pr_ptXI_SeaWa	= PR_PT XI_SEAWA(P,T,XI)	Prandtl Number	[-]	[1], [2], [3]	3/53
$Pr_l = f(t, \xi)$	Prl_tXI_SeaWa	= PRL_TXI_SEAWA(T,XI)	Prandtl Number of subcooled liquid	[-]	[1], [2], [3]	3/54
$Pr_{sl} = f(p_s, t_s, \xi_{sl})$	Prsl_pstsXisl_SeaWa	= PRSL_PST SXISL_SEAWA(PS,TS,XISL)	Prandtl Number of saturated liquid	[-]	[1], [2], [3]	3/55
$Pr_{sv} = f(p_s, t_s, \xi_{sl})$	Prsv_pstsXisl_SeaWa	= PRSV_PST SXISL_SEAWA(PS,TS,XISL)	Prandtl Number of saturated vapor	[-]	[1], [2], [3]	3/56
$\rho = f(p, t, \xi)$	rho_ptXI_SeaWa	= RHO_PT XI_SEAWA(P,T,XI)	Density	kg/m ³	[1], [2]	3/57
$\rho_l = f(p, t, \xi)$	rho_l_ptXI_SeaWa	= RHOL_PT XI_SEAWA(P,T,XI)	Density of subcooled liquid	kg/m ³	[1], [2]	3/58
$\rho_{sl} = f(p_s, t_s, \xi_{sl})$	rhosl_pstsXisl_SeaWa	= RHOSL_PST SXISL_SEAWA(PS,TS,XISL)	Density of saturated liquid	kg/m ³	[1], [2]	3/59
$\rho_{sv} = f(p_s, t_s, \xi_{sl})$	rhosv_pstsXisl_SeaWa	= RHOSV_PST SXISL_SEAWA(PS,TS,XISL)	Density of saturated vapor	kg/m ³	[1], [2]	3/60
$s = f(p, t, \xi)$	s_ptXI_SeaWa	= S_PT XI_SEAWA(P,T,XI)	Specific entropy	kJ/(kg*K)	[1], [2]	3/61
$s_l = f(p, t, \xi)$	sl_ptXI_SeaWa	= SL_PT XI_SEAWA(P,T,XI)	Specific entropy of subcooled liquid	kJ/(kg*K)	[1], [2]	3/62
$s_{sl} = f(p_s, t_s, \xi_{sl})$	ssl_pstsXisl_SeaWa	= SSL_PST SXISL_SEAWA(PS,TS,XISL)	Specific entropy of saturated liquid	kJ/(kg*K)	[1], [2]	3/63

Functional Dependence	Function Name in Excel®	Call as Fortran Program	Property or Function	Unit of the Result	Reference	Page
$s_{sv} = f(p_s, t_s, \xi_{sl})$	ssv_pstsXisl_SeaWa	= SSV_PSTSXISL_SEAWA(PS,TS,XISL)	Specific entropy of saturated vapor	kJ/(kg*K)	[1], [2]	3/64
$t_s = f(p_s, \xi_{sl})$	ts_psXisl_SeaWa	= TS_PSXISL_SEAWA(PS,XISL)	Boiling temperature	°C	[1], [2], [4]	3/65
$t_{mel} = f(p, \xi)$	tmel_pXi_SeaWa	= TMEL_PXI_SEAWA(P,XI)	Freezing temperature	°C	[1], [4], [5]	3/66
$t_{tr} = f(\xi)$	ttr_Xi_SeaWa	= TTR_XI_SEAWA(T,XI)	Triple point temperature	°C	[1], [4], [5]	3/67
Region = $f(p, t, \xi)$	Region_ptXi_SeaWa	= REGION_PTXI_SEAWA(P,T,XI)	Region	[-]	[1], [2], [4]	3/68
Region = $f(p, h, \xi)$	Region_phXi_SeaWa	= REGION_PHXI_SEAWA(P,H,XI)	Region	[-]	[1], [2], [4]	3/69
Region = $f(p, s, \xi)$	Region_psXi_SeaWa	= REGION_P SXI_SEAWA(P,S,XI)	Region	[-]	[1], [2], [4]	3/70
$t = f(p, h, \xi)$	t_phXi_SeaWa	= T_PHXI_SEAWA(P,H,XI)	Backward function: Temperature from pressure and specific enthalpy	°C	[1], [2], [4]	3/71
$t = f(p, s, \xi)$	t_psXi_SeaWa	= T_P SXI_SEAWA(P,S,XI)	Backward function: Temperature from pressure and specific entropy	°C	[1], [2], [4]	3/72
$u = f(p, t, \xi)$	u_ptXi_SeaWa	= U_PTXI_SEAWA(P,T,XI)	Specific internal energy	kJ/kg	[1], [2]	3/73
$u_l = f(p, t, \xi)$	ul_ptXi_SeaWa	= UL_PTXI_SEAWA(P,T,XI)	Specific internal energy of subcooled liquid	kJ/kg	[1], [2]	3/74
$u_{sl} = f(p_s, t_s, \xi_{sl})$	usl_pstsXisl_SeaWa	= USL_PSTSXISL_SEAWA(PS,TS,XISL)	Specific internal energy of saturated liquid	kJ/kg	[1], [2]	3/75
$u_{sv} = f(p_s, t_s, \xi_{sl})$	usv_pstsXisl_SeaWa	= USV_PSTSXISL_SEAWA(PS,TS,XISL)	Specific internal energy of saturated vapor	kJ/kg	[1], [2]	3/76
$v = f(p, t, \xi)$	v_ptXi_SeaWa	= V_PTXI_SEAWA(P,T,XI)	Specific volume	m³/kg	[1], [2]	3/77
$v_l = f(p, t, \xi)$	vl_ptXi_SeaWa	= VL_PTXI_SEAWA(P,T,XI)	Specific internal energy of subcooled liquid	m³/kg	[1], [2]	3/78
$v_{sl} = f(p_s, t_s, \xi_{sl})$	vsl_pstsXisl_SeaWa	= VSL_PSTSXISL_SEAWA(PS,TS,XISL)	Specific volume of saturated liquid	m³/kg	[1], [2]	3/79
$v_{sv} = f(p_s, t_s, \xi_{sl})$	vsv_pstsXisl_SeaWa	= VSV_PSTSXISL_SEAWA(PS,TS,XISL)	Specific volume of saturated vapor	m³/kg	[1], [2]	3/80

Functional Dependence	Function Name in Excel®	Call as Fortran Program	Property or Function	Unit of the Result	Reference	Page
$w = f(p, t, \xi)$	w_ptXI_SeaWa	= W_PTXI_SEAWA(P,T,XI)	Speed of sound	m/s	[1]	3/81
$w_l = f(p, t, \xi)$	wl_ptXI_SeaWa	= WL_PTXI_SEAWA(P,T,XI)	Speed of sound of liquid	m/s	[1]	3/82
$w_{sl} = f(p_s, t_s, \xi_{sl})$	wsl_pstsXisl_SeaWa	= WSL_PSTSXISL_SEAWA(PS,TS,XISL)	Speed of sound of saturated liquid	m/s	[1]	3/83
$w_{sv} = f(p_s, t_s, \xi_{sv})$	wsv_pstsXisl_SeaWa	= WSV_PSTSXISL_SEAWA(PS,TS,XISL)	Speed of sound of saturated vapor	m/s	[1]	3/84
$x = f(p, t, \xi)$	x_ptXI_SeaWa	= X_PTXI_SEAWA(P,T,XI)	Vapor fraction	kg/kg	[1]	3/85
$\xi_{sl} = f(p_s, t_s)$	Xisl_psts_SeaWa	= XISL_PSTS_SEAWA(PS,TS)	Mass fraction of sea salt in saturated liquid	kg/kg	[1]	3/86
$\xi_{sv} = f(p_s, t_s)$	Xisv_psts_SeaWa	= XISV_PSTS_SEAWA(PS,TS)	Mass fraction of sea salt in saturated vapor	kg/kg	[1]	3/87

Range of Validity

Pressure:	0.002 093	$\leq$	p	$\leq$	1000	bar
Temperature:	0	$\leq$	t	$\leq$	220	°C
Salinity:	0	$\leq$	ξ	$\leq$	0.2	kg _{salt} /kg _{mixture}

Reference State

Property	SeaWater
Pressure	1.01325 bar
Temperature	0 °C
Salinity	0.003516504 kg/kg
Enthalpy	0 kJ/ kg
Entropy	0 kJ/(kg K)

Variable Types for Function Call

All functions and variables:	REAL*8
------------------------------	--------

Property functions with deviation to the range of validity (cp. Chapter. 3)

2 Application of FluidMAT in Mathcad®

The Add-In FluidMAT SeaWa has been developed to calculate thermodynamic properties in Mathcad® more conveniently.

Within Mathcad it enables the direct call of functions relating to seawater from the LibSeaWa property library.

2.1 Installing FluidMAT LibSeaWa

In order to carry out the following steps it is compulsory that Mathcad® is already installed on your computer. Before you begin, it is best to close Mathcad.

Now, insert the CD FluidMAT and complete the following steps:

FluidMat will be installed by means of a self-extracting program. In order to run the installation click "Start" in the lower task bar of your desktop.

- Click "Systemsteuerung (Control Panel)"
- Double click "Software (Add or Remove Programs)"
- Click the button "CD oder Floppy (CD or Floppy)" in the "Eigenschaften von Software" (Add or Remove Programs) menu.

In the "Run Installation Program" dialog box, click the "Durchsuchen...(Browse...)" button and choose the file "FluidMAT_LibSeaWa_Setup.exe" on your CD. The path in the dialog box should be:

<partition>:\FluidMAT_LibSeaWa_Setup.exe ,

whereas <partition> is the name of the CD-Rom drive.

If you do not have the setup file on CD, search and start the file "FluidMAT_LibSeaWa_Setup.exe" in the specific folder on your hard disk drive by clicking the "Durchsuchen... (Browse...)" button.

- The installation begins when you click the "Ende > (Finish >)" button.

Now, a window appears noting that all Windows programs should be closed.

- When this is the case, the installation can be continued. Click the "Weiter > (Continue >)" button.
- Now, you will be informed on the FluidMAT product in the "Liesmich-Datei (ReadMe-File)" window. Click "Next >" to leave this window.

In the following dialog box, "Zielpfad wählen (Choose Destination Location)" (see figure 2.1), the default path where Mathcad has been installed will be shown

C:\Programme\Mathcad\Mathcad 14\ (for German version of Windows®)

C:\Program Files\Mathcad\Mathcad 14\ (for English version of Windows).

By clicking the "Verzeichnis wechseln... (Browse...)" button, you can change the installation directory before installation.

The path will be displayed in the window.

- Click on "Weiter > (Next >)" in the window "Zielpfad wählen"
- Click on the "Weiter > (Next >)" button in the "Installation beginnen" window

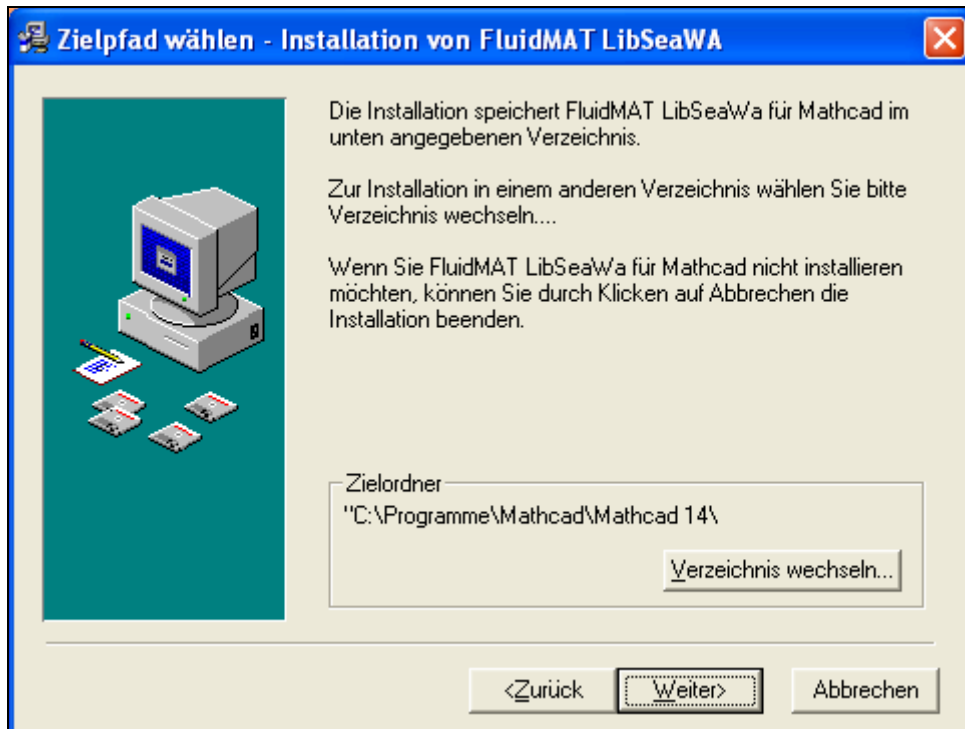


Figure 2.1: Zielpfad wählen (Choose Destination Location)

Depending on your Windows configuration, you may be warned that Windows must be restarted for the installation process. If this occurs, first remove the CD from the disk drive and then click "OK" for restart.

After rebooting Windows, start the installation again from the beginning.

After FluidMAT has been installed, the sentence "FluidMAT LibSeaWa wurde erfolgreich installiert. (FluidMAT has been successfully installed)" will be shown.

- Confirm this by clicking the "Fertigstellen (Finish)" button.

Close the "Eigenschaften von Software (Control Panel)" window. The installation of FluidMAT LibSeaWa has been carried out.

During the installation process the following files have been copied into the chosen destination folder (which is the folder, where Mathcad was installed):

LibSeaWa.dll	Property library for seawater
Readme_MAT_LibSeaWa.txt	ReadMe file.

The following file was installed into your Mathcad subdirectory \userEFI:

MAT_LibSeaWa.dll	Function definition of seawater
------------------	---------------------------------

The following file was installed into your Mathcad subdirectory \doc\funcdoc:

MAT_LibSeaWa_DE.xml	Function registration in the dialog window "Insert Function" for LibSeaWa (German Mathcad)
---------------------	--

Now, you have to overwrite the file "LibSeaWa.dll" in your Mathcad directory with the file of the same name provided on your CD with FluidMAT.

To do this, open the CD in "My Computer" and click on the file "LibSeaWa.dll" in order to highlight it. Then click on the "Edit" menu in your Explorer and select "Copy".

Now, open your Mathcad directory (the standard being C:\Program Files\Mathcad\Mathcad 14) and insert the file "LibSeaWa.dll" by clicking the "Edit" menu in your Explorer and then select "Paste". Answer the question whether you want to replace the file by clicking the "Yes" button. Now, you have overwritten the file "LibSeaWa.dll" successfully.

If you are using an English version of Windows it is necessary to copy the file "MAT_LibSeaWa_EN.xml" into the Mathcad subfolder funcdoc (the standard path being: C:\Program Files\Mathcad\Mathcad 14\doc\funcdoc).

To do this, open the CD in "My Computer" and click on the file " MAT_LibSeaWa_EN.xml" in order to highlight it. Then click on the "Edit" menu in your Explorer and select "Copy".

Now, open your Mathcad directory C:\Program Files\Mathcad\Mathcad 14\doc\funcdoc and insert the file "MAT_LibSeaWa_EN.xml" by clicking the "Edit" menu in your Explorer and then select "Paste".

Proceed in the same way to replace the file "MAT_LibSeaWa_DE.xml". Open the CD in "My Computer" and click on the file "MAT_LibSeaWa_DE.xml" in order to highlight it. Then click on the "Edit" menu in your Explorer and select "Copy".

Now, open your Mathcad directory (the standard being C:\Program Files\Mathcad\Mathcad 14\doc\funcdoc) and insert the file "MAT_LibSeaWa_DE.xml" by clicking the "Edit" menu in your Explorer and then select "Paste". Answer the question whether you want to replace the file by clicking the "Yes" button. Now, you have overwritten the file "MAT_LibSeaWa_DE.xml" successfully.

If you are using Mathcad in version 11 or lower, you have to rename the appropriate .xml file to "MAT_LibSeaWa.xml".

From within Mathcad you can now select the property functions.

2.2 Licensing the LibSeaWa Property Library

The licensing procedure has to be carried out when you are calculating a function with LibSeaWa in Mathcad® and a FluidMAT prompt message appears. In this case, you will see the "License Information" window for LibSeaWa (see Figure 2.2).

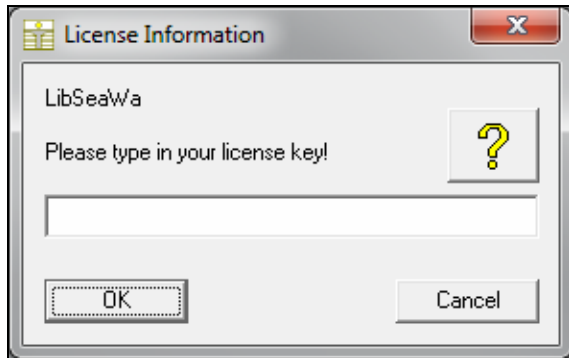


Figure 2.2: "License Information" window

Here you will have to type in the license key which you have obtained from the Zittau/Goerlitz University of Applied Sciences. You can find contact information on the "Content" page of this User's Guide or by clicking the yellow question mark in the "License Information" window. Then the following window will appear:

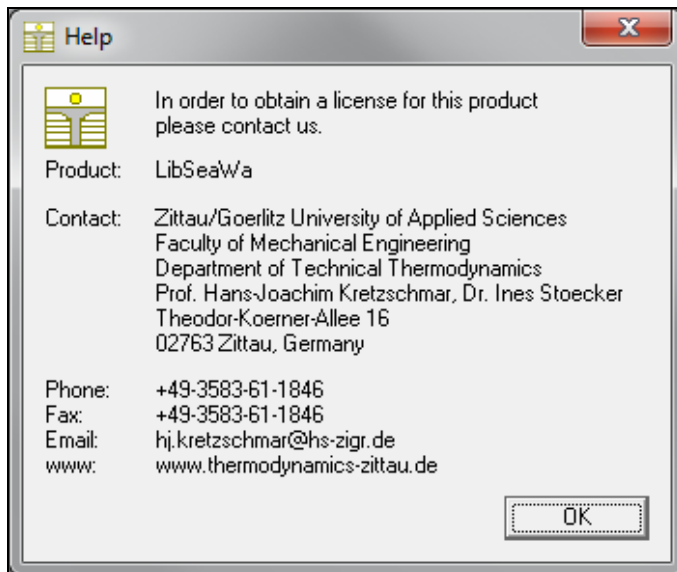


Figure 2.3: "Help" window

If you do not enter a valid license it is still possible to use Mathcad by clicking "Cancel". In this case, the LibSeaWa property library will display the result "-11111111" for every calculation.

The "License Information" window will appear every time you use the LibSeaWa property library in Mathcad unless you uninstall FluidMAT.

If you decide not to use FluidMAT LibSeaWa, you can uninstall the program following the instructions given in section 2.4 of this User's Guide.

2.3 Example: Calculation of $h = f(p, t, \xi)$

Now we will calculate, step by step, the specific enthalpy h as a function of pressure, temperature and mass fraction of sea salt for seawater, by using FluidMAT.

Please carry out the following steps:

- Start Mathcad.
- It is recommended to prepare a Mathcad sheet as shown in Figure 2.4:

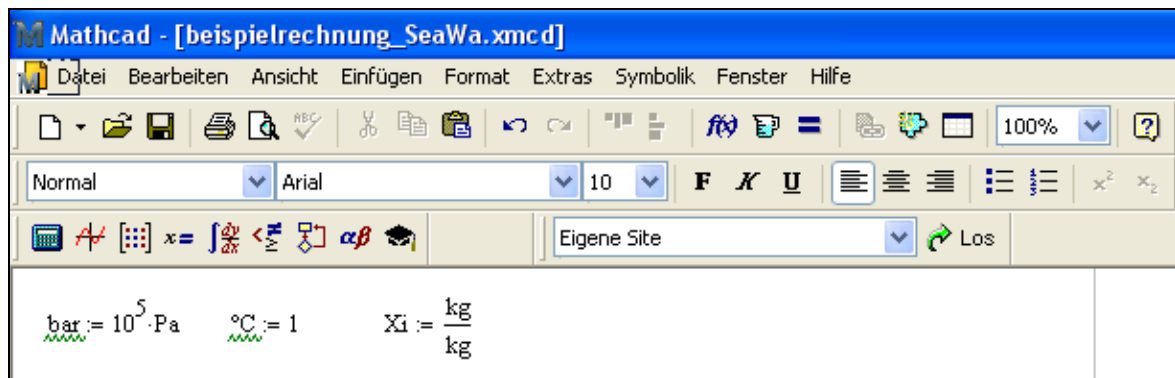


Figure 2.4: Example Mathcad sheet for the calculation with used units

Type "p:" and enter the value for pressure p in bar

(Range of validity of LibSeaWa: $p = 0.01 \text{ bar} \dots 1000 \text{ bar}$)

⇒ e.g.: Enter "p:1*bar"

- Type "t:" and enter the value for temperature t in °C

(Range of validity of LibSeaWa: $t = -6^\circ\text{C} \dots 220^\circ\text{C}$)

⇒ e.g.: Enter "t:100*°C"

- Type "Xi:" and enter the value for mass fraction of sea salt ξ in kg/kg

(Range of validity of LibSeaWa: $\xi = 0 \text{ kg/kg} \dots 0.12 \text{ kg/kg}$)

⇒ e.g.: Enter "Xi:0.12*kg/kg"

The Mathcad sheet should now look as shown in figure 2.5.

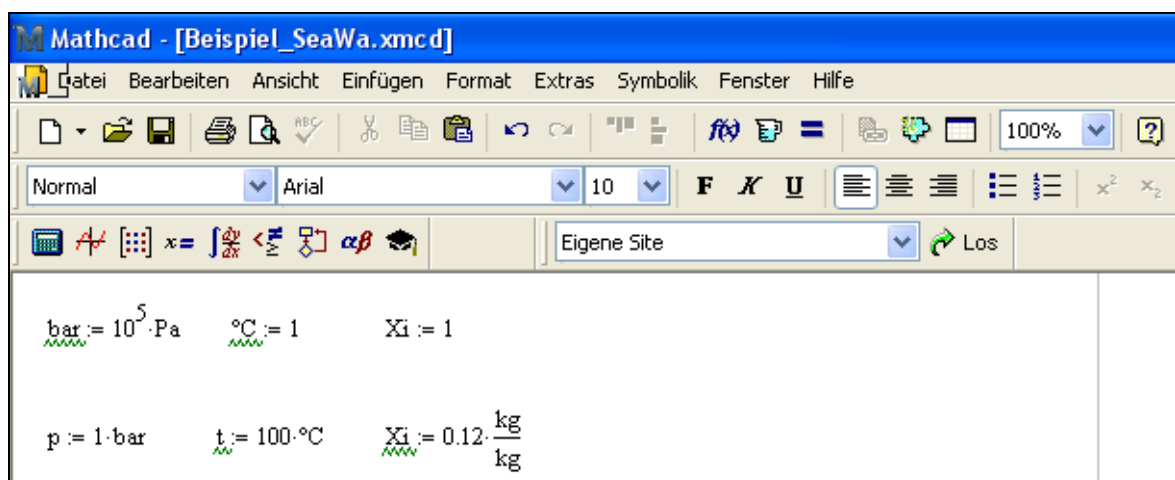


Figure 2.5: Example sheet after input of the given parameters

- Enter the symbol for the result and then a colon

⇒ e.g.: Type "h:" and press the Enter key

- Click "Einfügen (Insert)" in the Mathcad menu bar and then "Funktion... (Function...)"
The "Funktion einfügen (Insert function)" window appears (see figure 2.6)
- Click "LibSeaWa" under "Funktionskategorie: (Function category:)" on the left hand side (see Figure 2.4)
- Click "h_ptXi_SeaWa" under "Funktionsname: (Function name:)" on the right hand side

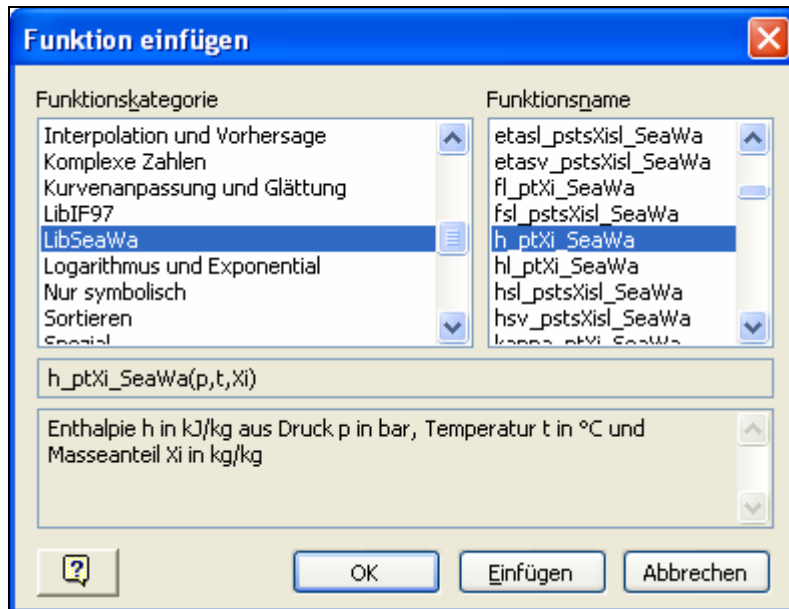


Figure 2.6: Choice of library and function name

- Click the "OK" button
Now you will see the line "**h_ptXi_SeaWa(■, ■, ■)**" in the Mathcad window (see figure 2.7).

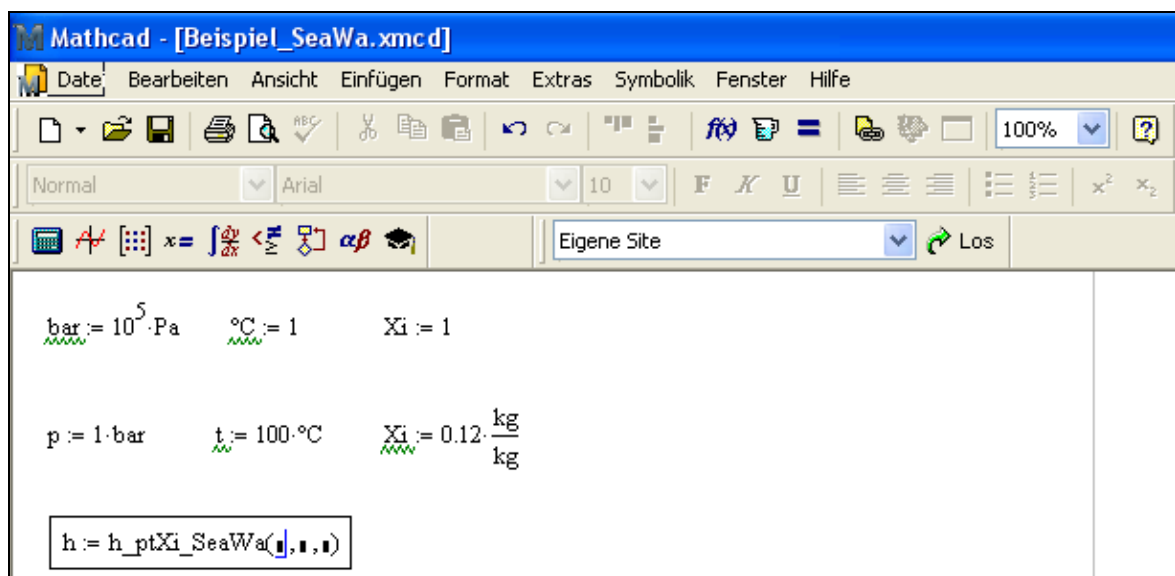


Figure 2.7: Example Mathcad sheet with formula and placeholders

- The cursor is now situated on the first operand. You can now enter the value for p either by entering the value directly or by entering the name of the variable where the value was saved.

⇒ e.g.: Enter "p/bar" because this property function works with this value only.

- Situate the cursor on the next placeholder. You can now enter the value for the temperature t either by entering the value for t directly or by typing the name of the variable in which the value of the temperature has been saved.

⇒ e.g.: Enter "t/°C" because this property function works with this value only.

- Situate the cursor on the next placeholder. You can now enter the value for the mass fraction of sea salt ζ either by entering the value for ζ directly or by typing the name of the variable in which the value of the mass fraction of sea salt has been saved.

⇒ e.g.: Enter "Xi / (kg/kg)" because this property function works with this value only.

- Close the input formula by pressing the "Enter"-Key.
- You can now go on working with the variable h which we have just calculated or you can query the result. If you wish to see the result, you have to type the command "**h=**" on the next line in the Mathcad window
- The result for h in kJ/kg appears.

⇒ The result in our sample calculation here is: $h = 359.287 \text{ kJ/kg}$.

The representation of the result depends on the amount of places and decimal places which you have set before.

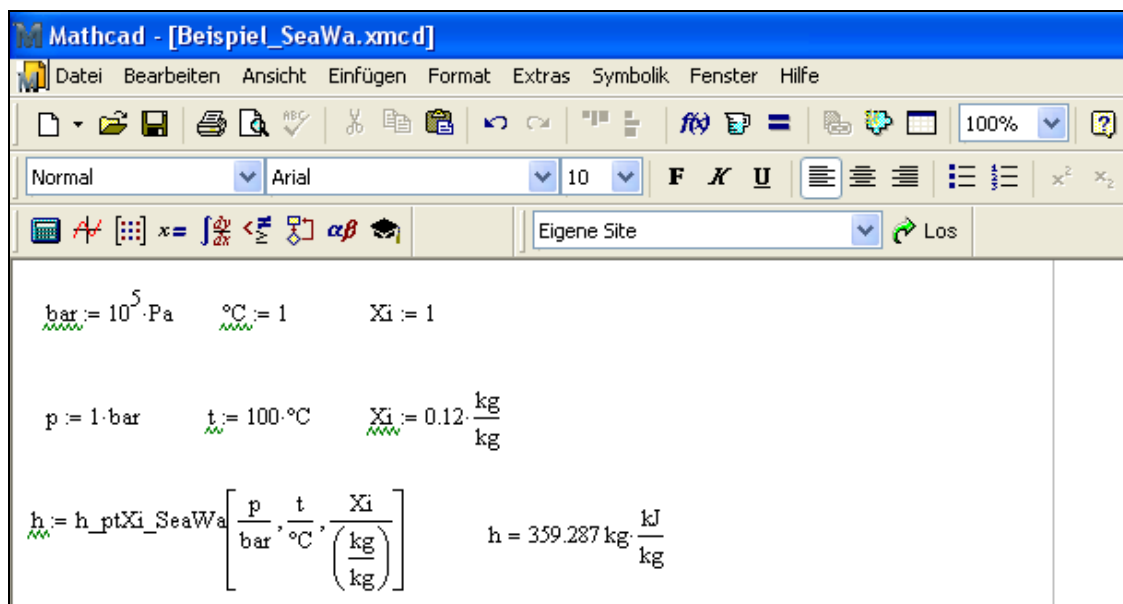


Figure 2.8: Example Mathcad sheet with finished calculation

The calculation of $h = f(p, t, \zeta)$ has been carried out.

You can now arbitrarily change the values for p , t , and ζ . The specific enthalpy is recalculated and updated every time you change the data. This shows that the Mathcad® data flow and the DLL calculations are working together successfully.

2.4 Removing FluidMAT LibSeaWa

To remove FluidMAT LibSeaWa from Mathcad[®] and your hard disk drive, carry out the following steps:

- Click the "Start" button in the Windows task bar
- Click "Einstellungen (Settings)"
- Click "Systemsteuerung (Control Panel)"
- Double click "Software (Add or Remove Programs)"
- Click on "FluidMAT SeaWa" in the list box
- Click the button "Hinzufügen/Entfernen... (Add or Remove)"
- Mark "Automatisch" and click the "Weiter > (Next >)" button
- Click "Ende (Finish)" in the "Perform Uninstall" window

Finally, close the "Eigenschaften von Software (Add or Remove Programs)" and "Systemsteuerung (Control Panel)" windows.

Now FluidMAT SeaWa has been removed.

3. Program Documentation

Thermal Diffusivity $a = f(p, t, \xi)$
Function Name:

a_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION A_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

a_ptXi_SeaWa, a - Thermal diffusivity in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 100 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Thermal diffusivity $a = \frac{\lambda}{\rho * c_p}$
- This function is not defined in the wet steam region.

Result for Wrong Input Values:

a_ptXi_SeaWa, $a = -1000$

References:

$a(p, t, \xi)$ [1], [2]

Thermal Diffusivity of Liquid Seawater $a_l = f(t, \xi)$
Function Name:

al_tXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION AL_TXI_SEAWA(T, XI), REAL*8 T, XI
```

Input Values:

t - Temperature t in °C

ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

al_tXi_SeaWa, a_l - Thermal diffusivity of liquid seawater in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C

Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Thermal diffusivity $a = \frac{\lambda}{\rho * c_p}$

Result for Wrong Input Values:

al_tXi_SeaWa, $a_l = -1000$

References:

$a(t, \xi)$ [1]

Thermal Diffusivity of Saturated Liquid $a_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

asl_pstsXisl_SeaWa

Fortran Programs:

REAL*8 FUNCTION ASL_PSTXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

asl_pstsXisl_SeaWa, asl - Thermal diffusivity of saturated seawater in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Thermal diffusivity $a = \frac{\lambda}{\rho * c_p}$

Possible input variants:

$$a^{sl} = f(-1000, t_s, \xi^{sl})$$

$$a^{sl} = f(p_s, -1000, \xi^{sl})$$

$$a^{sl} = f(p_s, t_s, -1000)$$

$$a^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

asl_pstsXisl_SeaWa, asl = -1000

References:

$a(p, t, \xi)$ [1],[2]

Thermal Diffusivity of Saturated Vapor $a_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

asv_pstsXisl_SeaWa

Fortran Programs:

REAL*8 FUNCTION ASV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS, TS,XISL

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

asv_pstsXisl_SeaWa, asv - Thermal diffusivity of saturated seawater in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Thermal diffusivity $a = \frac{\lambda}{\rho * c_p}$

Result for Wrong Input Values:

asv_pstsXisl_SeaWa, asv = -1000

References:

$a(p, t, \xi)$ [1],[2]

Thermal Expansion Coefficient of Liquid Seawater $\alpha_l = f(p, t, \xi)$
Function Name:

alpha_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION  ALPHAL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

alpha_ptXi_SeaWa, alpha - Thermal expansion of liquid seawater in 1/K

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ bar to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

alpha_ptXi_SeaWa, alpha = -1000

References:

$\alpha(p, t, \xi)$ [1]

Thermal Expansion Coefficient of Saturated Liquid $\alpha_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

alphasl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION  ALPHASL_PSTSXISL_SEAWA(PS, TS, XISL),
REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

alphasl_pstsXisl_SeaWa, alphasl - Thermal expansion of saturated seawater in 1/K

Range of Validity:

Temperature t : from 0 °C to 80 °C
Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to $p_s(t = 80\text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\alpha^{sl} = f(-1000, t_s, \xi^{sl})$$

$$\alpha^{sl} = f(p_s, -1000, \xi^{sl})$$

$$\alpha^{sl} = f(p_s, t_s, -1000)$$

$$\alpha^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

alphasl_pstsXisl_SeaWa, alphasl = -1000

References:

$\alpha(p, t, \xi)$ [1]

Haline Contraction Coefficient of Liquid Seawater $\beta_l = f(p, t, \xi)$
Function Name:

betal_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION BETAL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

betal_ptXi_SeaWa, betal - Haline contraction coefficient of liquid seawater in kg/kg

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ bar to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

betal_ptXi_SeaWa, betal = -1000

References:

$\beta(p, t, \xi)$ [1]

Haline Contraction Coefficient of Saturated Liquid $\beta_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

betasl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION BETASL_PSTXISL_SEAWA(PS, TS, XISL),
REAL*8 PS, TS, XISL
```

Input Values:

p_s - Pressure p in bar
 t_s - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

betasl_pstsXisl_SeaWa, betasl - Haline contraction coefficient of saturated seawater in 1/K

Range of Validity:

Temperature t : from 0 °C to 80 °C
Mixture pressure p : from $p_s (t = 0 \text{ °C}, \xi = 0.12)$ to $p_s (t = 80 \text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\beta^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \beta^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \beta^{sl} &= f(p_s, t_s, -1000) \\ \beta^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

betasl_ptXi_SeaWa, betasl = -1000

References:

$\beta(p, t, \xi)$ [1]

Isentropic Temperature - Pressure Coefficient (Adiabatic Lapse Rate) of Liquid Seawater $\beta_{\text{sl}} = f(p, t, \xi)$

Function Name:

betalsl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION BETAISL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

betalsl_ptXi_SeaWa, betalsl - adiabatic lapse rate in K/kPa

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ bar to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

betalsl_ptXi_SeaWa, betalsl = -1000

References:

$\beta_{\text{sl}}(p, t, \xi)$ [1]

Isentropic Temperature - Pressure Coefficient (Adiabatic Lapse Rate) of Saturated Liquid $\beta_{ls}^{sl} = f(p_s, t_s, \xi^{sl})$

Function Name:

betalssl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION BETAISL_PSTSL_SEAWA(PS, TS, XISL),
REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

betalssl_pstsXisl_SeaWa, betalssl - Adiabatic lapse rate of saturated seawater in K/kPa

Range of Validity:

Temperature t : from 0 °C to 80 °C
Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\beta_{ls}^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \beta_{ls}^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \beta_{ls}^{sl} &= f(p_s, t_s, -1000) \\ \beta_{ls}^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

betalssl_pstsXisl_SeaWa, betalssl = -1000

References:

$$\beta_{ls}(p, t, \xi) \quad [1]$$

Specific Isobaric Heat Capacity $c_p = f(p, t, \xi)$
Function Name:

cp_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION CP_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

cp_ptXi_SeaWa, c_p - Specific isobaric heat capacity in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ bar to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- This function is not defined in the wet steam region.

Result for Wrong Input Values:

cp_ptXi_SeaWa, $c_p = -1000$

References:

$c_p(p, t, \xi)$ [1], [2]

Specific Isobaric Heat Capacity of Liquid Seawater $c_{pI} = f(p, t, \xi)$
Function Name:

cpl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION CPL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

cpl_ptXi_SeaWa, cpl - Specific isobaric heat capacity of liquid seawater in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ bar to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

cpl_ptXi_SeaWa, cpl = -1000

References:

$c_p(p, t, \xi)$ [1], [2]

Specific Isobaric Heat Capacity of Saturated Liquid $c_{p_{sl}} = f(p_s, t_s, \xi^{sl})$

Function Name:

cpsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION CPSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

cpsl_pstsXisl_SeaWa, cpsl - Specific isobaric heat capacity of saturated seawater in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$c_p^{sl} = f(-1000, t_s, \xi^{sl})$$

$$c_p^{sl} = f(p_s, -1000, \xi^{sl})$$

$$c_p^{sl} = f(p_s, t_s, -1000)$$

$$c_p^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

cpsl_ptXi_SeaWa, cpsl = -1000

References:

$c_p(p, t, \xi)$ [1],[2]

Specific Isobaric Heat Capacity of Saturated Vapor $c_{p_{sv}} = f(p_s, t_s, \xi^{sl})$

Function Name:

cpsv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION  CPSV_PSTSXISL_SEAWA(PS,TS,XISL)  REAL*8 PS,TS,XISL
```

Input Values:

p_s - Pressure p in bar
 t_s - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

cpsv_pstsXisl_SeaWa, cpsv - Specific isobaric heat capacity of saturated seawater
in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

cpsv_pstsXisl_SeaWa, cpsv = -1000

References:

$c_p(p, t, \xi)$ [1], [2]

Dynamic Viscosity $\eta = f(p, t, \xi)$
Function Name:

eta_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION ETA_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

eta_ptXi_SeaWa, eta - Dynamic Viscosity in Pa*s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to 100 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- This function is not defined in the wet steam region.

Result for Wrong Input Values:

eta_ptXi_SeaWa, eta = -1000

References:

$\eta(p, t, \xi)$ [1],[2]

Dynamic Viscosity of Liquid Seawater $\eta_l = f(t, \xi)$
Function Name:

etal_tXi_SeaWa

Fortran Programs:

REAL*8 FUNCTION ETAL_PTXI_SEAWA(T, XI), REAL*8 T, XI

Input Values: t - Temperature t in °C Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture**Result:**

etal_tXi_SeaWa, etal - Dynamic Viscosity in Pa*s

Range of Validity:Temperature t : from 0 °C to 220 °CMass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture**Result for Wrong Input Values:**

etal_tXi_SeaWa, etal = -1000

References: $\eta(\rho, t, \xi)$ [1], [2]

Dynamic Viscosity of Saturated Liquid $\eta_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

etasl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION ETASL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

etasl_pstsXisl_SeaWa, etasl - Dynamic Viscosity in Pa*s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\eta^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \eta^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \eta^{sl} &= f(p_s, t_s, -1000) \\ \eta^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

etasl_ptXi_SeaWa, etasl = -1000

References:

$\eta(p, t, \xi)$ [1], [2]

Dynamic Viscosity of Saturated Vapor $\eta_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

etasv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION ETASV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

etasv_pstsXisl_SeaWa, etasv - Dynamic Viscosity of saturated seawater in Pa*s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

etasv_pstsXisl_SeaWa, etasv = -1000

References:

$\eta(p, t, \xi)$ [1], [2]

Specific Helmholtz Energy of Liquid Seawater $f_l = f(p, t, \xi)$

Function Name:

fl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION FL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

fl_ptXi_SeaWa, fl - Specific Helmholtz energy in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

fl_ptXi_SeaWa, fl = -1000

References:

$f(p, t, \xi)$ [1]

Specific Helmholtz Energy of Saturated Liquid $f_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

fsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION FSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

fsl_pstsXisl_SeaWa, fsl - Specific Helmholtz energy in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to $p_s(t = 80\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}
 f^{sl} &= f(-1000, t_s, \xi^{sl}) \\
 f^{sl} &= f(p_s, -1000, \xi^{sl}) \\
 f^{sl} &= f(p_s, t_s, -1000) \\
 f^{sl} &= f(p_s, t_s, \xi^{sl})
 \end{aligned}$$

Result for Wrong Input Values:

fsl_pstsXisl_SeaWa, fsl = -1000

References:

$f(p, t, \xi)$ [1]

Osmotic Coefficient of Liquid Seawater $\phi_l = f(p, t, \xi)$
Function Name:

phil_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PHIL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

phil_ptXi_SeaWa, phil - Osmotic Coefficient in [-]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 < \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

phil_ptXi_SeaWa, phil = -1000

References:

$\phi(p, t, \xi)$ [1]

Osmotic Coefficient of Saturated Liquid $\phi_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

phisl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PHISL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

phisl_pstsXisl_SeaWa, phisl - Osmotic Coefficient in [-]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 < \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\phi^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \phi^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \phi^{sl} &= f(p_s, t_s, -1000) \\ \phi^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

phisl_pstsXisl_SeaWa, fsl = - 1000

References:

$\phi(p, t, \xi)$ [1]

Specific Enthalpy $h = f(p, t, \xi)$

Function Name:

h_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION H_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

h_ptXi_SeaWa, h - specific enthalpy in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

h_ptXi_SeaWa, h = -1000

References:

$h(p, t, \xi)$ [1], [2]

Specific Enthalpy of Liquid Seawater $h_l = f(p, t, \xi)$

Function Name:

hl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION HL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

hl_ptXi_SeaWa, hl - Specific Enthalpy of liquid seawater in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

hl_ptXi_SeaWa, hl = -1000

References:

$h(p, t, \xi)$ [1], [2]

Specific Enthalpy of Saturated Fluid $h_{sl} = f(p_s, t_s, \xi^{sl})$

Function Name:

hsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION HSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

hsl_pstsXisl_SeaWa, hsl - Specific Enthalpy of saturated seawater in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$h^{sl} = f(-1000, t_s, \xi^{sl})$$

$$h^{sl} = f(p_s, -1000, \xi^{sl})$$

$$h^{sl} = f(p_s, t_s, -1000)$$

$$h^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

hsl_pstsXisl_SeaWa, hsl = -1000

References:

$h(p, t, \xi)$ [1] , [2]

Specific Enthalpy of Saturated Steam $h_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

hsv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION HSV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS, TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

hsv_pstsXisl_SeaWa, hsv - Specific Enthalpy of saturated seawater in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

hsv_pstsXisl_SeaWa, hsv = -1000

References:

$h(p, t, \xi)$ [1] , [2]

Isentropic Exponent $\kappa = f(p, t, \xi)$
Function Name:

kappa_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPA_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappa_ptXi_SeaWa, kappa - Isentropic Exponent in [-]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- Isentropic Exponent $\kappa = \frac{w^2}{p * v}$
- This function is not defined in the wet steam region.

Result for Wrong Input Values:

kappa_ptXi_SeaWa, kappa = -1000

References:

$\kappa(p, t, \xi)$ [1]

Isentropic Exponent of Liquid Seawater $\kappa_i = f(p, t, \xi)$
Function Name:

kappal_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPAL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappal_ptXi_SeaWa, kappal - Isentropic Exponent of liquid seawater in [-]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- Isentropic Exponent $\kappa = \frac{w^2}{p * v}$

Result for Wrong Input Values:

kappal_ptXi_SeaWa, kappal = -1000

References:

$\kappa(p, t, \xi)$ [1]

Isentropic Exponent of Saturated Fluid $\kappa_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

kappasl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPASL_PSTSL_XISL_SEAWA(PS, TS, XISL)
REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappasl_pstsXisl_SeaWa, kappasl - Isentropic Exponent of saturated seawater in [-]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- Isentropic Exponent $\kappa = \frac{w^2}{p * v}$

Possible input variants:

$$\begin{aligned}\kappa^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \kappa^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \kappa^{sl} &= f(p_s, t_s, -1000) \\ \kappa^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

kappasl_pstsXisl_SeaWa, kappasl = -1000

References:

$\kappa(p, t, \xi)$ [1]

Isentropic Exponent of Saturated Vapor $\kappa_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

kappasv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPASV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

p_s - Pressure p in bar
 t_s - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappasv_pstsXisl_SeaWa, kappasv - Isentropic Exponent of saturated seawater in [-]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- Isentropic Exponent $\kappa = \frac{w^2}{p \cdot v}$

Result for Wrong Input Values:

kappasv_pstsXisl_SeaWa, kappasv = -1000

References:

$\kappa(p, t, \xi)$ [1]

Isothermal Compressibility of Liquid Seawater $\kappa_{\text{TI}} = f(p, t, \xi)$
Function Name:

kappaTI_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPATL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappaTI_ptXi_SeaWa, kappaTI - Isothermal Compressibility of liquid seawater in [1/kPa]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

kappaTI_ptXi_SeaWa, kappaTI = -1000

References:

$\kappa_{\text{T}}(p, t, \xi)$ [1]

Isothermal Compressibility of Saturated Fluid $\kappa_{T\text{sl}} = f(p_s, t_s, \xi^{\text{sl}})$
Function Name:

kappaTsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPATSL_PSTSXISL_SEAWA(PS, TS, XISL)
REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappaTsl_pstsXisl_SeaWa, kappaTsl - Isothermal Compressibility of saturated fluid in [1/kPa]

Range of Validity

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to $p_s(t = 80\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\kappa_T^{\text{sl}} &= f(-1000, t_s, \xi^{\text{sl}}) \\ \kappa_T^{\text{sl}} &= f(p_s, -1000, \xi^{\text{sl}}) \\ \kappa_T^{\text{sl}} &= f(p_s, t_s, -1000) \\ \kappa_T^{\text{sl}} &= f(p_s, t_s, \xi^{\text{sl}})\end{aligned}$$

Result for Wrong Input Values:

kappaTsl_pstsXisl_SeaWa, kappaTsl = -1000

References:

$\kappa_T(p, t, \xi)$ [1]

Isothermal Compressibility of Liquid Seawater $\kappa_{\text{ISL}} = f(p, t, \xi)$
Function Name:

kappaIsl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPAISL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappaIsl_ptXi_SeaWa, kappaIsl - Isentropic Compressibility of liquid seawater in [1/kPa]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

kappaIsl_ptXi_SeaWa, kappaIsl = -1000

References:
 $\kappa_{\text{IS}}(p, t, \xi)$ [1]

Isothermal Compressibility of Saturated Seawater $\kappa_{Is\,sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

kappalssl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION KAPPAISL_PSTSL_SEAWA(PS, TS, XISL)
REAL*8 PS, TS, XISL
```

Input Values:

p_s - Pressure p in bar
 t_s - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

kappalssl_pstsXisl_SeaWa, kappalssl - Isentropic Compressibility of saturated seawater in [1/kPa]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to $p_s(t = 80\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\kappa_{Is}^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \kappa_{Is}^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \kappa_{Is}^{sl} &= f(p_s, t_s, -1000) \\ \kappa_{Is}^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

kappalssl_pstsXisl_SeaWa, kappalssl = -1000

References:

$\kappa_{Is}(p, t, \xi)$ [1]

Thermal Conductivity $\lambda = f(p, t, \xi)$
Function Name:

lambda_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION LAMBDA_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

lambda_ptXi_SeaWa, lambda - Thermal Conductivity in W/(m*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 100 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- This function is not defined in the wet steam region.

Result for Wrong Input Values:

lambda_ptXi_SeaWa, lambda = -1000

References:

$\lambda(p, t, \xi)$ [3], [4], [15]

Thermal Conductivity of Liquid Seawater $\lambda_l = f(t, \xi)$
Function Name:

lambdal_tXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION LAMBDAL_TXI_SEAWA(T, XI), REAL*8 T, XI
```

Input Values:

t - Temperature t in °C

Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

lambdal_tXi_SeaWa, lambdal - Thermal Conductivity of liquid seawater in W/(m*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C

Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

lambdal_tXi_SeaWa, lambdal = -1000

References:

$\lambda(\rho, t, \xi)$ [3], [4], [15]

Thermal Conductivity of Saturated Fluid $\lambda_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

lambdasl_pstsXisl_SeaWa

Fortran Programs:

```

      REAL*8 FUNCTION  LAMBDA_SL_PSTSL_SEAWA(PS, TS, XISL)
      REAL*8 PS, TS, XISL
      ps      - Pressure  $p$  in bar
      ts      - Temperature  $t$  in °C
      Xisl    - Mass fraction of sea salt  $\xi$  in kg sea salt/kg mixture
  
```

Result:

lambdasl_pstsXisl_SeaWa, lambdasl - Thermal Conductivity of saturated fluid in W/(m*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$\lambda^{sl} = f(-1000, t_s, \xi^{sl})$$

$$\lambda^{sl} = f(p_s, -1000, \xi^{sl})$$

$$\lambda^{sl} = f(p_s, t_s, -1000)$$

$$\lambda^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

lambdasl_pstsXisl_SeaWa, lambdasl = -1000

References:

$\lambda(p, t, \xi)$ [3], [4], [15]

Thermal Conductivity of Saturated Vapor $\lambda_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

lambdasv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION LAMBDAV_PSTSL_SEAWA(PS,TS,XISL)
REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

lambdasvXisl_psts_SeaWa, lambdasv - Thermal Conductivity of saturated seawater in W/(m*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

lambdasv_pstsXisl_SeaWa, lambdasv = -1000

References:

$\lambda(p, t, \xi)$ [3], [4], [15]

Relative Chemical Potential of Liquid Seawater $\mu_l = f(p, t, \xi)$
Function Name:

myl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION MYL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

myl_ptXi_SeaWa, myl - Relative chemical potential of liquid seawater in [kJ/kg]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

myl_ptXi_SeaWa, myl = -1000

References:

$\mu(p, t, \xi)$ [1]

Relative Chemical Potential of Saturated Fluid $\mu_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

mysl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION MYSL_PSTXISL_SEAWA(PS, TS, XISL)
REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
ts - Temperature t in °C
Xisl - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

mysl_pstsXisl_SeaWa, mysl - Relative chemical potential of saturated seawater in [kJ/kg]

Range of Validity:

Temperature t : from 0 °C to 80 °C
Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\mu^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \mu^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \mu^{sl} &= f(p_s, t_s, -1000) \\ \mu^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

mysl_pstsXisl_SeaWa, mysl = -1000

References:

$$\mu(p, t, \xi) \quad [1]$$

Chemical Potential of H₂O of Liquid Seawater $\mu_{wl} = f(p, t, \xi)$
Function Name:

mywl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION MYWL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

mywl_ptXi_SeaWa, mywl - Chemical potential of H₂O of liquid seawater in [kJ/kg]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

mywl_ptXi_SeaWa, mywl = -1000

References:

$\mu_w(p, t, \xi)$ [1]

Chemical Potential of H₂O of Saturated Fluid $\mu_{w\,sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

mywsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION MYWSL_PSTSXISL_SEAWA(PS, TS, XISL)
REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
ts - Temperature t in °C
Xisl - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

mywsl_pstsXisl_SeaWa, mywsl - Relative chemical potential of H₂O of saturated seawater in [kJ/kg]

Range of Validity:

Temperature t : from 0 °C to 80 °C
Mixture pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.12)$ to $p_s(t = 80\text{ °C}, \xi = 0)$
Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\mu_w^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \mu_w^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \mu_w^{sl} &= f(p_s, t_s, -1000) \\ \mu_w^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

mywsl_pstsXisl_SeaWa, mywsl = -1000

References:

$$\mu_w(p, t, \xi) \quad [1]$$

Chemical Potential of Sea Salt of Liquid Seawater $\mu_{\text{Salt I}} = f(p, t, \xi)$
Function Name:

mySaltI_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION MYSALTPTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

mySaltI_ptXi_SeaWa, mySaltI - Chemical potential of sea salt of liquid seawater
in [kJ/kg]

Range of Validity:

Temperature t : from 0 °C to 80 °C
Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

mySaltI_ptXi_SeaWa, mySaltI = -1000

References:

$\mu_S(p, t, \xi)$ [1]

Chemical Potential of Sea Salt of Saturated Liquid $\mu_{\text{Salt sl}} = f(p_s, t_s, \xi^{\text{sl}})$
Function Name:

mySaltsI_pstsXisl_SeaWa

Fortran Programs:

```

      REAL*8 FUNCTION  MYSALTSL_PSTSLXISL_SEAWA(PS, TS, XISL)
      REAL*8 PS, TS, XISL
      ps      - Pressure p in bar
      ts      - Temperature t in °C
      Xisl    - Mass fraction of sea salt  $\xi$  in kg sea salt/kg mixture
  
```

Result:

mySaltsI_pstsXisl_SeaWa, mySaltsI - Relative chemical potential of H₂O of saturated seawater in [kJ/kg]

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned} \mu_{\text{Salt}}^{\text{sl}} &= f(-1000, t_s, \xi^{\text{sl}}) \\ \mu_{\text{Salt}}^{\text{sl}} &= f(p_s, -1000, \xi^{\text{sl}}) \\ \mu_{\text{Salt}}^{\text{sl}} &= f(p_s, t_s, -1000) \\ \mu_{\text{Salt}}^{\text{sl}} &= f(p_s, t_s, \xi^{\text{sl}}) \end{aligned}$$

Result for Wrong Input Values:

mySaltsI_pstsXisl_SeaWa, mySaltsI = -1000

References:

$$\mu_S(p, t, \xi) \quad [1]$$

Kinematic Viscosity $\nu = f(p, t, \xi)$
Function Name:

ny_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION NY_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ny_ptXi_SeaWa, ny - Kinematic Viscosity in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 100 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Kinematic Viscosity $\nu = \frac{\eta}{\rho}$
- This function is not defined in the wet steam region.

Result for Wrong Input Values:

ny_ptXi_SeaWa, ny = -1000

References:

$\nu(p, t, \xi)$ [1], [2]

Kinematic Viscosity of Liquid Seawater $\nu_l = f(t, \xi)$
Function Name:

nyl_tXi_SeaWa

Fortran Programs:

REAL*8 FUNCTION NYL_TXI_SEAWA(T, XI), REAL*8 T, XI

Input Values:
 t - Temperature t in °C

 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

 nyl_tXi_SeaWa, nyl - Kinematic Viscosity of liquid seawater in m²/s

Range of Validity:

 Temperature t : from 0 °C to 220 °C

 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

 - Kinematic Viscosity $\nu = \frac{\eta}{\rho}$
Result for Wrong Input Values:

nyl_tXi_SeaWa, nyl = -1000

References:
 $\nu(\rho, t, \xi)$ [1],[2]

Kinematic Viscosity of Saturated Liquid $\nu_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

nysl_pstsXisl_SeaWa

Fortran Programs:

REAL*8 FUNCTION NYSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

nysl_pstsXisl_SeaWa, nysl - Kinematic Viscosity of saturated seawater in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Kinematic Viscosity $\nu = \frac{\eta}{\rho}$

Possible input variants:

$$\begin{aligned}\nu^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \nu^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \nu^{sl} &= f(p_s, t_s, -1000) \\ \nu^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

nysl_pstsXisl_SeaWa, nysl = -1000

References:

$\nu(p, t, \xi)$ [1],[2]

Kinematic Viscosity of Saturated Vapor $\nu_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

nysv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION ASV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

nysv_pstsXisl_SeaWa, nysv - Kinematic Viscosity of saturated seawater in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Kinematic Viscosity $\nu = \frac{\eta}{\rho}$

Result for Wrong Input Values:

nysv_pstsXisl_SeaWa, nysv = -1000

References:

$\nu(p, t, \xi)$ [1],[2]

Boiling Pressure $p_s = f(t_s, \xi^{sl})$
Function Name:

ps_tsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PS_TSXISL_SEAWA( TS, XISL), REAL*8 TS, XISL
```

Input Values:

t_s - Saturated Temperature t_s in °C

X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ps_tsXisl_SeaWa, ps - Saturation Pressure in bar

Range of Validity:

Temperature t : from 0 °C to 220 °C

Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

ps_tsXisl_SeaWa, ps = -1000

References:

$p_s(t_s, \xi_{sl})$ [1], [2],[4]

Freezing Pressure $p_{\text{mel}} = f(t, \xi)$
Function Name:

pmel_tXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PMEL_TXI_SEAWA( T, XI), REAL*8 T, XI
```

Input Values:

t - Temperature t in °C

Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

pmel_tXi_SeaWa, pmel - Melting Pressure in bar

Range of Validity:

Temperature t : from 0 °C to 0.01 °C

Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- If $p_{\text{mel}} > 1000$ bar then pmel_tXi_SeaWa, pmel = -1000
- If $p_{\text{mel}} < p_{\text{tr}}(\xi)$ then pmel_tXi_SeaWa, pmel = -1000

Result for Wrong Input Values:

pmel_tXi_SeaWa, pmel = -1000

References:

$p_{\text{mel}}(t, \xi)$ [1], [4], [5]

Triple Point Pressure $p_{tr} = f(\xi)$
Function Name:

ptr_Xi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PTR_XI_SEAWA( XI), REAL*8 XI
```

Input Values:

ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ptr_Xi_SeaWa, ptr - Triplepoint Pressure in bar

Range of Validity:

Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

ptr_Xi_SeaWa, ptr = -1000

References:

$p_{tr}(\xi)$ [1], [4], [5]

Prandtl Number $Pr = f(p, t, \xi)$
Function Name:

Pr_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PR_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Pr_ptXi_SeaWa, Pr - Prandtl Number in [-]

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 100 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Prandtl Number $Pr = \frac{V}{a}$
- This function is not defined in the wet steam region.

Result for Wrong Input Values:

Pr_ptXi_SeaWa, Pr = -1000

References:

$Pr(p, t, \xi)$ [1], [2], [3]

Prandtl Number of Liquid Seawater $Pr_l = f(t, \xi)$

Function Name:

Prl_tXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PRL_TXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Prl_ptXi_SeaWa, Prl - Prandtl Number of liquid seawater in [-]

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Prandtl Number $Pr = \frac{V}{a}$

Result for Wrong Input Values:

Prl_tXi_SeaWa, Prl = -1000

andtl Number **References:**

$Pr(p, t, \xi)$ [1], [2], [3]

Prandtl Number of Saturated Liquid $Pr_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

Prsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PRSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Prsl_pstsXisl_SeaWa, Prsl - Prandtl Number of saturated seawater in [-]

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$Pr^{sl} = f(-1000, t_s, \xi^{sl})$$

$$Pr^{sl} = f(p_s, -1000, \xi^{sl})$$

$$Pr^{sl} = f(p_s, t_s, -1000)$$

$$Pr^{sl} = f(p_s, t_s, \xi^{sl})$$

Comments:

- Prandtl Number $Pr = \frac{\nu}{a}$

Result for Wrong Input Values:

Prsl_ptXi_SeaWa, Prsl = -1000

References:

$Pr(p, t, \xi)$ [1], [2], [3]

Prandtl Number of Saturated Vapor $Pr_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

Prsv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION PRSV_PSTSXISL_SEAWA(PS,TS), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Prsv_pstsXisl_SeaWa, Prsv - Prandtl Number of saturated seawater in m²/s

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Prandtl Number $Pr = \frac{\nu}{a}$

Result for Wrong Input Values:

Prsv_pstsXisl_SeaWa, Prsv = -1000

References:

$Pr(p, t, \xi)$ [1], [2], [3]

Density $\rho = f(p, t, \xi)$
Function Name:

rho_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION RHO_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

rho_ptXi_SeaWa, rho - Density in kg/m³

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

rho_ptXi_SeaWa, rho = -1000

References:

$\rho(p, t, \xi)$ [1], [2]

Density of Liquid Seawater $\rho_l = f(p, t, \xi)$
Function Name:

rhoI_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION RHOL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 X_i - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

rhoI_ptXi_SeaWa, rhoI - Density of liquid seawater in kg/m³

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

rhoI_ptXi_SeaWa, rhoI = -1000

References:

$\rho(p, t, \xi)$ [1], [2], [3]

Density of Saturated Liquid $\rho_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

rhosl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION RHOSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

rhosl_pstsXisl_SeaWa, rhosl - Density of saturated seawater in kg/m³

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}\rho^{sl} &= f(-1000, t_s, \xi^{sl}) \\ \rho^{sl} &= f(p_s, -1000, \xi^{sl}) \\ \rho^{sl} &= f(p_s, t_s, -1000) \\ \rho^{sl} &= f(p_s, t_s, \xi^{sl})\end{aligned}$$

Result for Wrong Input Values:

rhosl_pstsXisl_SeaWa, rhosl = -1000

References:

$\rho(p, t, \xi)$ [1], [2], [3]

Density of Saturated Vapor $\rho_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

rhosv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION RHOSV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

rhosv_pstsXisl_SeaWa, rhosv - Density of saturated seawater in kg/m³

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

rhosv_pstsXisl_SeaWa, rhosv = -1000

References:

$\rho(p, t, \xi)$ [1], [2], [3]

Specific Entropy $s = f(p, t, \xi)$

Function Name:

s_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION S_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

s_ptXi_SeaWa, s - specific entropy in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

s_ptXi_SeaWa, $s = -1000$

References:

$s(p, t, \xi)$ [1], [2]

Specific Entropy of Liquid Seawater $s_l = f(p, t, \xi)$
Function Name:

sl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION SL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

sl_ptXi_SeaWa, sl - Specific Entropy of liquid seawater in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

sl_ptXi_SeaWa, sl = -1000

References:

$s(p, t, \xi)$ [1], [2]

Specific Entropy of Saturated Liquid $s_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

ssl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION  SSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ssl_pstsXisl_SeaWa, ssl - Specific Entropy of saturated seawater in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$s^{sl} = f(-1000, t_s, \xi^{sl})$$

$$s^{sl} = f(p_s, -1000, \xi^{sl})$$

$$s^{sl} = f(p_s, t_s, -1000)$$

$$s^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

ssl_pstsXi_SeaWa, ssl = -1000

References:

$s(p, t, \xi)$ [1] , [2]

Specific Entropy of Saturated Vapor $s_{sv} = f(p_s, t_s, \xi^{sl})$

Function Name:

ssv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION SSV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ssv_pstsXisl_SeaWa, ssv - Specific Entropy of saturated seawater in kJ/(kg*K)

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

ssv_pstsXisl_SeaWa, ssv = -1000

References:

$s(p, t, \xi)$ [1] , [2]

Boiling Temperature $t_s = f(p_s, \xi^{sl})$
Function Name:

ts_psXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION TS_PSXISL_SEAWA( PS, XISL), REAL*8 PS, XISL
```

Input Values:

ps - Pressure p in bar

$Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ts_psXisl_SeaWa, ts - Saturation Temperature in °C

Range of Validity:

Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to $p_s(t = 220\text{ °C}, \xi = 0)$

Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

ts_psXisl_SeaWa, ts = -1000

References:

$t_s(p_s, \xi_{sl})$ [1], [2], [4]

Freezing Temperature $t_{\text{mel}} = f(p, \xi)$
Function Name:

tmel_pXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION TMEL_PXI_SEAWA( P, XI), REAL*8 P, XI
```

Input Values:

p - Pressure p in bar

ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

tmel_pXi_SeaWa, tmel - Melting Temperature in °C

Range of Validity:

Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar

Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- If $t_{\text{mel}}(p, \xi) < -12.15 \text{ °C}$ then tmel_pXi_SeaWa, tmel = -1000

Result for Wrong Input Values:

tmel_pXi_SeaWa, tmel = -1000

References:

$t_{\text{mel}}(p, \xi)$ [1], [4], [5]

Triple Point Temperature $t_{tr} = f(\xi)$
Function Name:

ttr_Xi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION TTR_XI_SEAWA( XI), REAL*8 XI
```

Input Values:

Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ttr_Xi_SeaWa, ttr - Triplepoint Temperature in °C

Range of Validity:

Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Result for Wrong Input Values:

ttr_Xi_SeaWa, ttr = -1000

References:

$t_{tr}(\xi)$ [1], [4], [5]

Region $Region = f(p, t, \xi)$
Function Name:

Region_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION REGION_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Region_ptXi_SeaWa, Region - Region in [-]

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Region : 1 subcooled liquid calculated from IAPWS Industrial Formulation 2013
- 2 wet steam region calculated from IAPWS Industrial Formulation 2013
- 3 superheated steam calculated from IAPWS Industrial Formulation 2013
- 11 subcooled liquid calculated from Fichtner-Handbook
- 22 wet steam region calculated from Fichtner-Handbook
- 33 superheated steam calculated from Fichtner-Handbook

Result for Wrong Input Values:

Region_ptXi_SeaWa, Region = -1000

References:

Region(p, t, ξ) [1], [2], [4]

Region $Region = f(p, h, \xi)$
Function Name:

Region_phXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION REGION_PHXI_SEAWA(P, H, XI), REAL*8 P, H, XI
```

Input Values:

p - Pressure p in bar
 h - Specific Enthalpy h in kJ/kg
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Region_phXi_SeaWa, Region - Region in [-]

Range of Validity:

Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Enthalpy h : from h_{min} kJ/kg to h_{max} kJ/kg
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Region : 1 subcooled liquid calculated from IAPWS Industrial Formulation 2013
- 2 wet steam region calculated from IAPWS Industrial Formulation 2013
- 3 superheated steam calculated from IAPWS Industrial Formulation 2013
- 11 subcooled liquid calculated from Fichtner-Handbook
- 22 wet steam region calculated from Fichtner-Handbook
- 33 superheated steam calculated from Fichtner-Handbook

Result for Wrong Input Values:

Region_phXi_SeaWa, Region = -1000

References:

$Region(p, h, \xi)$ [1], [2], [4]

Region *Region* = $f(p, s, \xi)$
Function Name:

Region_psXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION REGION_PSI_SEAWA(P, S, XI), REAL*8 P, S, XI
```

Input Values:

p - Pressure p in bar
 s - Specific Entropy s in kJ/(kg*K)
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

Region_psXi_SeaWa, Region - Region in [-]

Range of Validity:

Pressure p : from $p_s(t = 0^\circ\text{C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Entropy s : from s_{min} kJ/(kg K) to s_{max} kJ/(kg K)
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Region : 1 subcooled liquid calculated from IAPWS Industrial Formulation 2013
- 2 wet steam region calculated from IAPWS Industrial Formulation 2013
- 3 superheated steam calculated from IAPWS Industrial Formulation 2013
- 11 subcooled liquid calculated from Fichtner-Handbook
- 22 wet steam region calculated from Fichtner-Handbook
- 33 superheated steam calculated from Fichtner-Handbook

Result for Wrong Input Values:

Region_psXi_SeaWa, Region = -1000

References:

$Region(p, s, \xi)$ [1], [2], [4]

Backward Function: Temperature $t = f(p, h, \xi)$
Function Name:

t_phXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION T_PHXI_SEAWA(P, H, XI), REAL*8 P, H, XI
```

Input Values:

p - Pressure p in bar
 h - Specific Enthalpy h in kJ/kg
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

t_phXi_SeaWa, t - Temperature in °C

Range of Validity:

Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Enthalpy h : from h_{min} kJ/kg to h_{max} kJ/kg
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

t_phXi_SeaWa, t = -1000

References:

$t(p, h, \xi)$ [1], [2], [4]

Backward Function: Temperature $t = f(p, s, \xi)$
Function Name:

t_psXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION T_PSXI_SEAWA(P, S, XI), REAL*8 P, S, XI
```

Input Values:

p - Pressure p in bar
 s - Specific Entropy h in kJ/(kg*K)
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

t_psXi_SeaWa, t - Temperature in °C

Range of Validity:

Pressure p : from $p_s(t = 0\text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Entropy s : from s_{min} kJ/(kg K) to s_{max} kJ/(kg K)
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

t_psXi_SeaWa, $t = -1000$

References:

$t(p, s, \xi)$ [1], [2], [4]

Specific Internal Energy $u = f(p, t, \xi)$
Function Name:

u_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION U_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 ξ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

u_ptXi_SeaWa, u - Specific internal energy in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Specific internal energy $u = h - (p * v)$

Result for Wrong Input Values:

u_ptXi_SeaWa, $u = -1000$

References:

$u(p, t, \xi)$ [1], [2]

Specific Internal Energy of Liquid Seawater $u_l = f(p, t, \xi)$
Function Name:

ul_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION UL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

ul_ptXi_SeaWa, ul - Specific internal energy of liquid seawater in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Specific internal energy $u = h - (p * v)$

Result for Wrong Input Values:

ul_ptXi_SeaWa, ul = -1000

References:

$u(p, t, \xi)$ [1] , [2]

Specific Internal Energy of Saturated Liquid $u_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

usl_pstsXisl_SeaWa

Fortran Programs:

REAL*8 FUNCTION USL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

usl_pstsXisl_SeaWa, usl - Specific internal energy of saturated seawater in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$\begin{aligned}
 u^{sl} &= f(-1000, t_s, \xi^{sl}) \\
 u^{sl} &= f(p_s, -1000, \xi^{sl}) \\
 u^{sl} &= f(p_s, t_s, -1000) \\
 u^{sl} &= f(p_s, t_s, \xi^{sl})
 \end{aligned}$$

Comments:

- Specific internal energy $u = h - (p * v)$

Result for Wrong Input Values:

usl_pstsXisl_SeaWa, usl = -1000

References:
 $u(p, t, \xi)$ [1] , [2]

Specific Internal Energy of Saturated Vapor $u_{sv} = f(p_s, t_s, \xi^{sl})$

Function Name:

usv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION USV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

usv_pstsXisl_SeaWa, usv - Specific internal energy of saturated seawater in kJ/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- Specific internal energy $u = h - (p * v)$

Result for Wrong Input Values:

usv_pstsXisl_SeaWa, usv = -1000

References:

$u(p, t, \xi)$ [1] , [2]

Specific Volume $v = f(p, t, \xi)$
Function Name:

v_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION V_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

v_ptXi_SeaWa, v - Specific volume in m³/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

v_ptXi_SeaWa, $v = -1000$

References:

$v(p, t, \xi)$ [1], [2]

Specific Volume of Liquid Seawater $v_l = f(p, t, \xi)$
Function Name:

vl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION VL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

vl_ptXi_SeaWa, vl - Specific volume of liquid seawater in m³/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

vl_ptXi_SeaWa, vl = -1000

References:

$v(p, t, \xi)$ [1] , [2]

Specific Volume of Saturated Liquid $v_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

vsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION VSL_PSTXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

vsl_pstsXisl_SeaWa, vsl - Specific volume of saturated seawater in m³/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Possible input variants:

$$v^{sl} = f(-1000, t_s, \xi^{sl})$$

$$v^{sl} = f(p_s, -1000, \xi^{sl})$$

$$v^{sl} = f(p_s, t_s, -1000)$$

$$v^{sl} = f(p_s, t_s, \xi^{sl})$$

Result for Wrong Input Values:

vsl_pstsXisl_SeaWa, vsl = -1000

References:

$v(p, t, \xi)$ [1] , [2]

Specific Volume of Saturated Vapor $v_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

vsv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION VSV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

vsv_pstsXisl_SeaWa, vsv - Specific volume of saturated seawater in m³/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Result for Wrong Input Values:

vsv_pstsXisl_SeaWa, vsv = -1000

References:

$v(p, t, \xi)$ [1] , [2]

Speed of Sound $w = f(p, t, \xi)$
Function Name:

w_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION W_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

w_ptXi_SeaWa, w - Speed of Sound in m/s

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- This function is not defined in the wet steam region.
- If the input values are outside the range of validity of [1] the result is –1000 (for high salinity, high pressures and temperatures above 65 °C)

Result for Wrong Input Values:

w_ptXi_SeaWa, $w = -1000$

References:

$w(p, t, \xi)$ [1]

Speed of Sound of Liquid $w_l = f(p, t, \xi)$
Function Name:

wl_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION WL_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

wl_ptXi_SeaWa, wl - Speed of Sound of liquid in m/s

Range of Validity:

Temperature t :	from 0 °C to 80 °C
Pressure p :	from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
Mass fraction ξ :	$0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- If the input values are outside the range of validity of [1] the result is –1000 (for high salinity, high pressures and temperatures above 65 °C)

Result for Wrong Input Values:

wl_ptXi_SeaWa, wl = –1000

References:

$w(p, t, \xi)$ [1]

Speed of Sound of Saturated Liquid $w_{sl} = f(p_s, t_s, \xi^{sl})$
Function Name:

wsl_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION WSL_PSTSXISL_SEAWA(PS, TS, XISL), REAL*8 PS, TS, XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

wsl_pstsXisl_SeaWa, wsl - Sound speed of saturated seawater in m/s

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Possible input variants:

$$w^{sl} = f(-1000, t_s, \xi^{sl})$$

$$w^{sl} = f(p_s, -1000, \xi^{sl})$$

$$w^{sl} = f(p_s, t_s, -1000)$$

$$w^{sl} = f(p_s, t_s, \xi^{sl})$$

Comments:

- If the input values are outside the range of validity of [1] the result is -1000 (for high salinity, high pressures and temperatures above 65 °C)

Result for Wrong Input Values:

wsl_pstsXisl_SeaWa, wsl = -1000

References:

$w(p, t, \xi)$ [1]

Speed of Sound of Saturated Vapor $w_{sv} = f(p_s, t_s, \xi^{sl})$
Function Name:

wsv_pstsXisl_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION WSV_PSTSXISL_SEAWA(PS,TS,XISL), REAL*8 PS,TS,XISL
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C
 $Xisl$ - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

wsv_pstsXisl_SeaWa, wsv - Specific volume of saturated seawater in m³/kg

Range of Validity:

Temperature t : from 0 °C to 80 °C
 Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.12)$ to $p_s(t = 80 \text{ °C}, \xi = 0)$
 Mass fraction ξ : $0 \leq \xi \leq 0.12$ kg sea salt/kg mixture

Comments:

- If the input values are outside the range of validity of [1] the result is –1000 (for high salinity, high pressures and temperatures above 65 °C)

Result for Wrong Input Values:

wsv_pstsXisl_SeaWa, wsv = –1000

References:

$w(p, t, \xi)$ [1]

Vapor Fraction $x = f(p, t, \xi)$
Function Name:

x_ptXi_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION X_PTXI_SEAWA(P, T, XI), REAL*8 P, T, XI
```

Input Values:

p - Pressure p in bar
 t - Temperature t in °C
 Xi - Mass fraction of sea salt ξ in kg sea salt/kg mixture

Result:

x_ptXi_SeaWa, x - Vapor Fraction in kg/kg

Range of Validity:

Temperature t : from 0 °C to 220 °C
 Pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to 1000 bar and $p \geq p_{\text{mel}}(t, \xi)$
 Mass fraction ξ : $0 \leq \xi \leq 0.2$ kg sea salt/kg mixture

Comments:

- This function is defined in the wet steam region.

Result for Wrong Input Values:

x_ptXi_SeaWa, $x = -1000$

References:

$x(p, t, \xi)$ [1]

Mass Fraction of Sea Salt of Saturated Liquid $\xi_{sl} = f(p_s, t_s)$
Function Name:

Xisl_psts_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION XISL_PSTS_SEAWA(PS, TS), REAL*8 PS, TS
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C

Result:

Xisl_psts_SeaWa, Xisl - Mass fraction of sea salt of saturated liquid
in kg sea salt/kg mixture

Range of Validity:

Temperature t : from 0 °C to 220 °C
Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$

Result for Wrong Input Values:

Xisl_psts_SeaWa, Xisl = -1000

References:

$\xi_{sl}(p_s, t_s)$ [1]

Mass Fraction of Sea Salt of Saturated Vapor $\xi_{sv} = f(p_s, t_s)$
Function Name:

Xisv_psts_SeaWa

Fortran Programs:

```
REAL*8 FUNCTION XISV_PSTS_SEAWA(PS, TS), REAL*8 PS, TS
```

Input Values:

ps - Pressure p in bar
 ts - Temperature t in °C

Result:

Xisv_psts_SeaWa, Xisv - Mass fraction of sea salt of saturated steam
in kg sea salt/kg mixture

Range of Validity:

Temperature t : from 0 °C to 220 °C
Mixture pressure p : from $p_s(t = 0 \text{ °C}, \xi = 0.2)$ to $p_s(t = 220 \text{ °C}, \xi = 0)$

Result for Wrong Input Values:

Xisv_psts_SeaWa, Xisv = -1000

References:

$\xi_{sv}(p_s, t_s)$ [1]



4. Property Libraries for Calculating Heat Cycles, Boilers, Turbines, and Refrigerators

Water and Steam

Library LibIF97

- Industrial Formulation IAPWS-IF97 (Revision 2007)
- Supplementary Standards
 - IAPWS-IF97-S01
 - IAPWS-IF97-S03rev
 - IAPWS-IF97-S04
 - IAPWS-IF97-S05
- IAPWS Revised Advisory Note No. 3 on Thermodynamic Derivatives (2008)

Humid Combustion Gas Mixtures

Library LibHuGas

Model: Ideal mixture of the real fluids:
 CO_2 - Span and Wagner O_2 - Schmidt and Wagner
 H_2O - IAPWS-95 Ar - Tegeler et al.
 N_2 - Span et al.
 and of the ideal gases:
 SO_2 , CO, Ne (Scientific Formulation of Bucker et al.)
 Consideration of:
 Dissociation from VDI 4670 and Poynting effect

Humid Air

Library LibHuAir

Model: Ideal mixture of the real fluids:
 • Dry Air from Lemmon et al.
 • Steam, water and ice from IAPWS-IF97 and IAPWS-06
 Consideration of:
 • Condensation and freezing of steam
 • Dissociation from the VDI 4670
 • Poynting effect from ASHRAE RP-1485

Carbon Dioxide including Dry Ice

Library LibCO2

Formulation of Span and Wagner (1994)

Seawater

Library LibSeaWa

IAPWS Formulation 2008 of Feistel and IAPWS-IF97

Ice

Library LibICE

Ice from IAPWS-06, Melting and sublimation pressures from IAPWS-08, Water from IAPWS-IF97, Steam from IAPWS-95 and -IF97

Ideal Gas Mixtures

Library LibIdGasMix

Model: Ideal mixture of the ideal gases:

Ar	NO	He	Propylene
Ne	H_2O	F_2	Propane
N_2	SO_2	NH_3	Iso-Butane
O_2	H_2	Methane	n-Butane
CO	H_2S	Ethane	Benzene
CO_2	OH	Ethylene	Methanol

 Air

Consideration of:
 • Dissociation from the VDI Guideline 4670

Library LibIDGAS

Model: Ideal gas mixture from VDI Guideline 4670

Consideration of:
 • Dissociation from the VDI Guideline 4670

Dry Air including Liquid Air

Library LibRealAir

Formulation of Lemmon et al. (2000)

Nitrogen

Library LibN2

Formulation of Span et al. (2000)

Hydrogen

Library LibH2

Formulation of Leachman et al. (2007)

Refrigerants

Ammonia

Library LibNH3

Formulation of Tillner-Roth (1995)

R134a

Library LibR134a

Formulation of Tillner-Roth and Baehr (1994)

Iso-Butane

Library LibButane_Iso

Formulation of Bucker et al. (2003)

n-Butane

Library LibButane_n

Formulation of Bucker et al. (2003)

Mixtures for Absorption Processes

Ammonia/Water Mixtures

Library LibAmWa

IAPWS Guideline 2001 of Tillner-Roth and Friend (1998)
 Helmholtz energy equation for the mixing term
 (also useable for calculating Kalina Cycle)

Water/Lithium Bromide Mixtures

Library LibWaLi

Formulation of Kim and Infante Ferreira (2004)
 Gibbs energy equation for the mixing term

Liquid Coolants

Liquid Secondary Refrigerants

Library LibSecRef

Liquid solutions of water with

$\text{C}_2\text{H}_6\text{O}_2$	Ethylene glycol
$\text{C}_3\text{H}_8\text{O}_2$	Propylene glycol
$\text{C}_2\text{H}_5\text{OH}$	Ethyl alcohol
CH_3OH	Methyl alcohol
$\text{C}_3\text{H}_8\text{O}_3$	Glycerol
K_2CO_3	Potassium carbonate
CaCl_2	Calcium chloride
MgCl_2	Magnesium chloride
NaCl	Sodium chloride
$\text{C}_2\text{H}_3\text{KO}_2$	Potassium acetate

 Formulation of the International Institute of Refrigeration (1997)

Siloxanes as ORC Working Fluids

Octamethylcyclotetrasiloxane $C_8H_{24}O_4Si_4$ **Library LibD4**

Decamethylcyclopentasiloxane $C_{10}H_{30}O_5Si_5$ **Library LibD5**

Tetradecamethylhexasiloxane $C_{14}H_{42}O_6Si_6$ **Library LibMD4M**

Hexamethyldisiloxane $C_6H_{18}OSi_2$ **Library LibMM**

Formulation of Colonna et al. (2006)

Dodecamethylcyclohexasiloxane $C_{12}H_{36}O_6Si_6$ **Library LibD6**

Decamethyltetrasiloxane $C_{10}H_{30}O_3Si_4$ **Library LibMD2M**

Dodecamethylpentasiloxane $C_{12}H_{36}O_4Si_5$ **Library LibMD3M**

Octamethyltrisiloxane $C_8H_{24}O_2Si_3$ **Library LibMDM**

Formulation of Colonna et al. (2008)

Propane

Library LibPropane

Formulation of Lemmon et al. (2007)

Methanol

Library LibCH3OH

Formulation of de Reuck and Craven (1993)

Ethanol

Library LibC2H5OH

Formulation of Schroeder et al. (2012)

Helium

Library LibHe

Formulation of Arp et al. (1998)

Hydrocarbons

Decane $C_{10}H_{22}$ **Library LibC10H22**

Isopentane C_5H_{12} **Library LibC5H12_ISO**

Neopentane C_5H_{12} **Library LibC5H12_NEO**

Isohexane C_6H_{14} **Library LibC5H14**

Toluene C_7H_8 **Library LibC7H8**

Formulation of Lemmon and Span (2006)

Further Fluids

Carbon monoxide CO **Library LibCO**

Carbonyl sulfide COS **Library LibCOS**

Hydrogen sulfide H_2S **Library LibH2S**

Dinitrogen monoxide N_2O **Library LibN2O**

Sulfur dioxide SO_2 **Library LibSO2**

Acetone C_3H_6O **Library LibC3H6O**

Formulation of Lemmon and Span (2006)

For more information please contact:

Zittau/Goerlitz University of Applied Sciences
Department of Technical Thermodynamics
Professor Hans-Joachim Kretzschmar
Dr. Ines Stoecker

Theodor-Koerner-Allee 16
02763 Zittau, Germany

Internet: www.thermodynamics-zittau.de

E-mail: hj.kretzschmar@hs-zigr.de

Phone: +49-3583-61-1846

Fax.: +49-3583-61-1846

The following thermodynamic and transport properties can be calculated^a:

Thermodynamic Properties

- Vapor pressure p_s
- Saturation temperature T_s
- Density ρ
- Specific volume v
- Enthalpy h
- Internal energy u
- Entropy s
- Exergy e
- Isobaric heat capacity c_p
- Isochoric heat capacity c_v
- Isentropic exponent κ
- Speed of sound w
- Surface tension σ

Transport Properties

- Dynamic viscosity η
- Kinematic viscosity ν
- Thermal conductivity λ
- Prandtl-number Pr

Backward Functions

- $T, v, s(p, h)$
- $T, v, h(p, s)$
- $p, T, v(h, s)$
- $p, T(v, h)$
- $p, T(v, u)$

Thermodynamic Derivatives

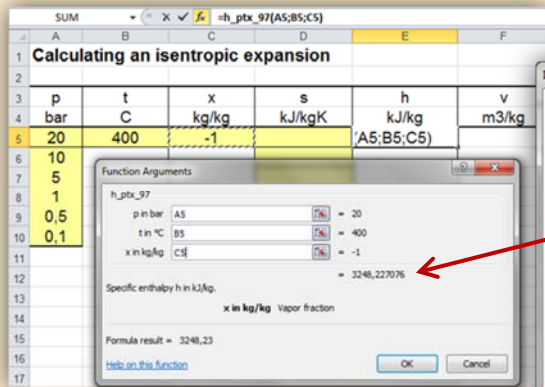
- Partial derivatives can be calculated.

^a Not all of these property functions are available in all property libraries.

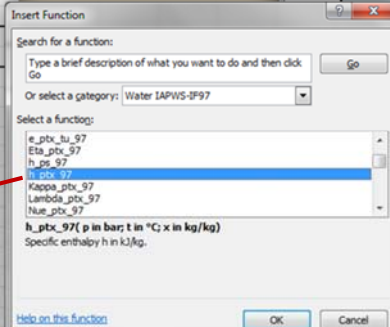


Property Software for Calculating Heat Cycles, Boilers, Turbines, and Refrigerators

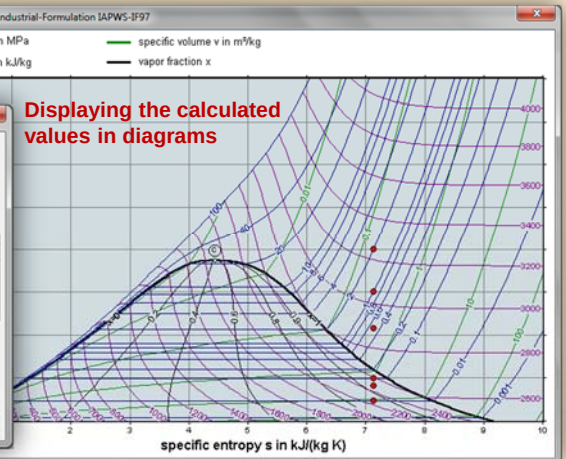
Add-In FluidEXL^{Graphics} for Excel[®]



Choosing a property library and a function



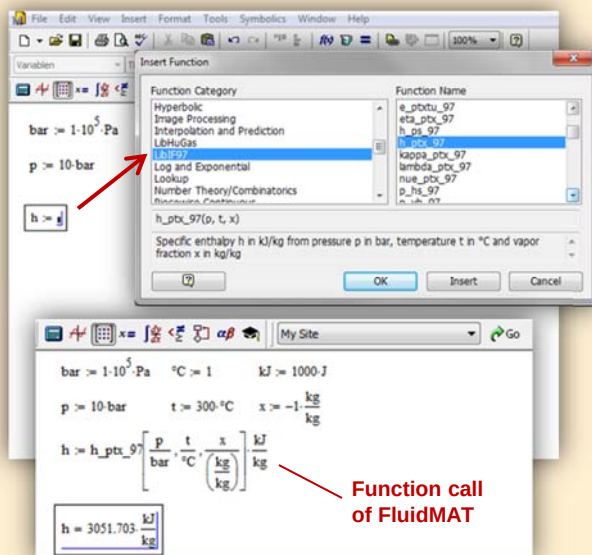
Displaying the calculated values in diagrams



Menu for the input of given property values

Add-In FluidMAT for Mathcad[®]

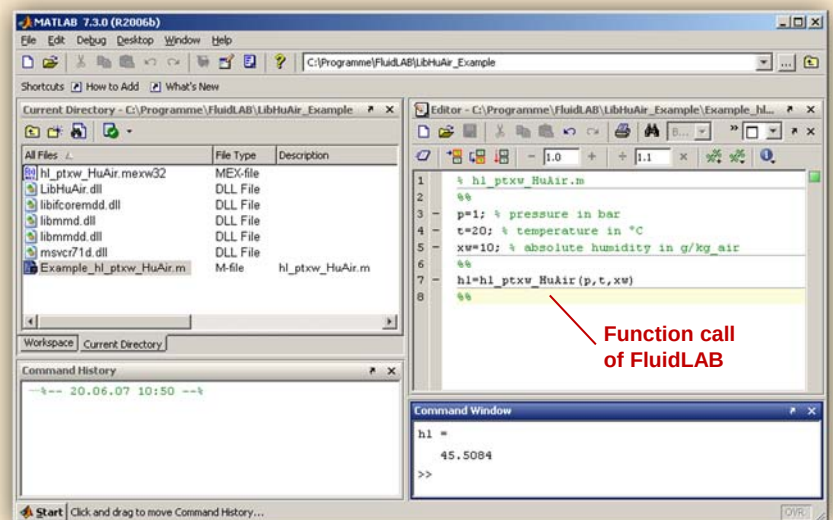
The property libraries can be used in Mathcad[®].



Function call of FluidMAT

Add-In FluidLAB for MATLAB[®]

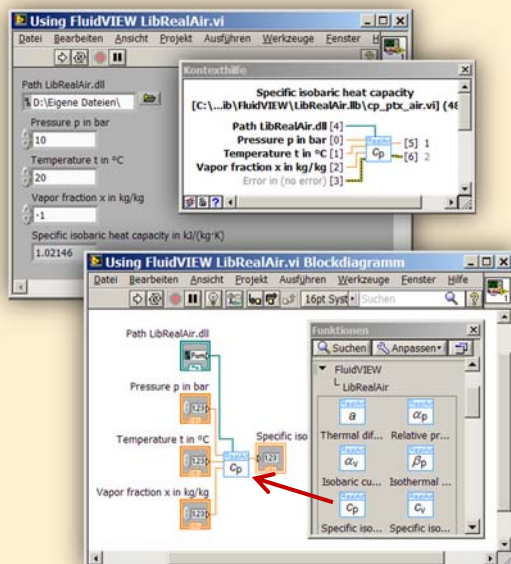
Using the Add-In FluidLAB the property functions can be called in MATLAB[®].



Function call of FluidLAB

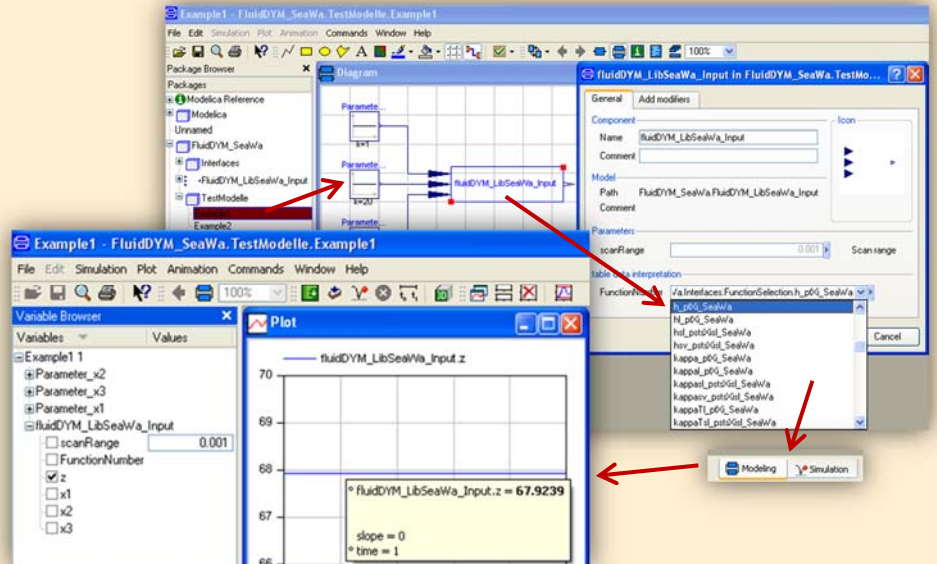
Add-On FluidVIEW for LabVIEW[®]

The property functions can be calculated in LabVIEW[®].

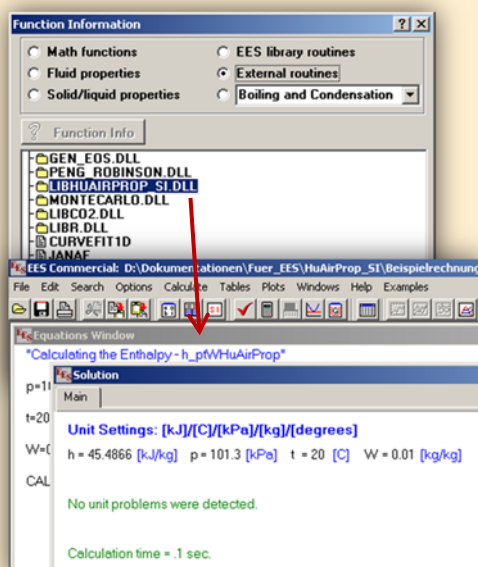


Add-In FluidDYM for DYMOLA[®] (Modelica) and SimulationX[®]

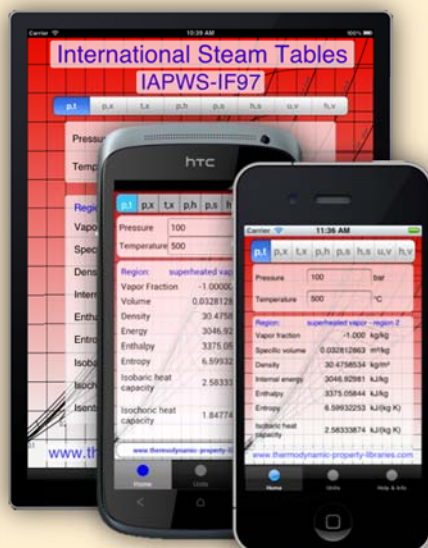
The property functions can be called in DYMOLA[®] and SimulationX[®].



Add-In FluidEES for Engineering Equation Solver®



App International Steam Tables for iPhone, iPad, iPod touch, Android smart phones and tablets



Online Property Calculator at www.thermodynamics-zittau.de

Zittau's Fluid Property Calculator

Fluid:

Function:

Unit System:

Enter given values: [Range of validity](#)

Pressure p: bar

Temperature t: °C

Vapor fraction x: kg/kg

Calculate / Recalculate

Result:

Specific enthalpy h = 3097.38 kJ/kg

For further information on property libraries available for EXCEL®, MATLAB®, Mathcad®, Engineering Equation Solver®, DYMOLA® (Modelica), SimulationX®, and LabView® click [here](#)

An App for calculating steam properties on iPhone, iPad, and iPod touch can be found [here](#)

PDF with the [description](#)

© Zittau/Goerlitz University of Applied Sciences
Faculty of Mechanical Engineering
Department of Technical Thermodynamics
Prof. Hans-Joachim Kretzschmar
Dr. Ines Stoecker
Programmer: Joachim Posselt

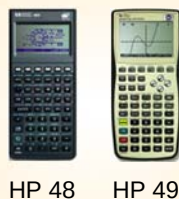
Tel.: +49-3583-61-1846 or -1881
Fax: +49-3583-61-1845
E-mail: info@thermodynamics-zittau.de
www.thermodynamics-zittau.de
www.thermodynamic-property-libraries.com
www.international-steam-tables.com
www.thermodynamic-formelsammlung.de

Property Software for Pocket Calculators

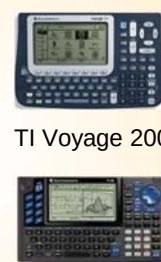
FluidCasio



FluidHP



FluidTI



TI Voyage 200

For more information please contact:

Zittau/Goerlitz University of Applied Sciences
Department of Technical Thermodynamics
Professor Hans-Joachim Kretzschmar
Dr. Ines Stoecker
Theodor-Koerner-Allee 16
02763 Zittau, Germany

E-mail: hj.kretzschmar@hs-zigr.de
Internet: www.thermodynamics-zittau.de
Phone: +49-3583-61-1846
Fax: +49-3583-61-1846

The following thermodynamic and transport properties^a can be calculated in Excel®, MATLAB®, Mathcad®, Engineering Equation Solver® EES, DYMOLA® (Modelica), SimulationX®, and LabVIEW®:

Thermodynamic Properties

- Vapor pressure p_s
- Saturation temperature T_s
- Density ρ
- Specific volume v
- Enthalpy h
- Internal energy u
- Entropy s
- Exergy e
- Isobaric heat capacity c_p
- Isochoric heat capacity c_v
- Isentropic exponent κ
- Speed of sound w
- Surface tension σ

Transport Properties

- Dynamic viscosity η
- Kinematic viscosity ν
- Thermal conductivity λ
- Prandtl-number Pr

Backward Functions

- $T, v, s(p, h)$
- $T, v, h(p, s)$
- $p, T, v(h, s)$
- $p, T(v, h)$
- $p, T(v, u)$

Thermodynamic Derivatives

- Partial derivatives can be calculated.

^a Not all of these property functions are available in all property libraries.

5. References

- [1] IAPWS Advisory Note No. 5: Industrial Calculation of the Thermodynamic Properties of Seawater. Available from www.iapws.org (2013)
- [2] Hömig, H.E:
Seawater an Seawater Distillation. Fichtner-Handbook,
Vulkan-Verlag, Essen (1978)
- [3] Jamieson, D.T; Tudhope, J. S.:
Physical Properties of Sea Water Solutions: Thermal Conductivity
Desalination 8, 393-401 (1970)
- [4] Feistel, R.:
A Gibbs Function for Seawater Thermodynamics for -6°C to 80°C and Salinity
up to 120g/kg. Deep Sea Research (2008)
- [5] IAPWS. Release on the IAPWS Formulation 1995 for the Thermodynamic
Properties of Ordinary Water Substances for General and Scientific Use.
Available from www.iapws.org (1995)
- [6] IAPWS. Release on an Equation of State for H₂O Ice Ih.
Available from www.iapws.org (2006)
- [7] IAPWS. Release on Revised Advisory Note No.3, Thermodynamic Derivatives
from IAPWS Formulations.
Available from www.iapws.org (2008)
- [8] IAPWS. Revised Release on the IAPWS Industrial Formulation 1997 for the
Thermodynamic Properties of Water and Steam IAPWS-IF97.
Available from www.iapws.org (2007)
- [9] Wagner, W.; Kretzschmar, H.-J.:
International Steam Tables.Springer-Verlag (2008)
- [10] Hellriegel, T.:
ITERA_PRO Iterationsroutine.
Fachgebiet Technische Thermodynamik, Hochschule Zittau/Görlitz (2008)
- [11] Miyagawa, K.:
IAPWS-IF97 for the Water Part of the EOS of SeaWater. (2008)
- [12] Kretzschmar, H.-J.; Stöcker, I.; Jähne, I.: Programm FluidEXL^{Graphics} für
Excel[®] Hochschule Zittau/Görlitz (FH), Fachbereich Maschinenwesen (1998 -
2008)
- [13] Salomo, B.:
Berechnung der thermodynamischen Stoffeigenschaften von
Kohlendioxid/Wasser-Gemischen für die Modellierung von Energie-
umwandlungsprozessen mit CO₂-Abscheidung Anbindung an MATLAB.
Diplomarbeit, Zittau (2007)
- [14] IAPWS. Revised Release on the Pressure along the Melting and Sublimation
Curves of Ordinary Water Substance. Available from www.iapws.org (2008)

6. Satisfied Customers

Date: 10/2011

The following companies and institutions use the property libraries

- FluidEXL^{Graphics} for Excel[®]
- FluidLAB for MATLAB[®]
- FluidMAT for Mathcad[®]
- FluidEES for Engineering Equation Solver[®] EES
- FluidDYM for Dymola[®] (Modelica)
- FluidVIEW for LabVIEW[®]:

2011

Lopez, Munguia, Spain	10/2011
University of KwaZulu-Natal, Westville, South Africa	10/2011
Voith, Heidenheim	09/2011
SpgBe Montreal, Canada	09/2011
SPG TECH, Montreuil Cedex, France	09/2011
Voith, Heidenheim-Mergelstetten	09/2011
MTU Aero Engines, Munich	08/2011
MIBRAG, Zeitz	08/2011
RWE, Essen	07/2011
Fels, Elingerode	07/2011
Weihenstephan University of Applied Sciences	07/2011, 09/2011, 10/2011
Forschungszentrum Juelich	07/2011
RWTH Aachen University	07/2011, 08/2011
INNEO Solutions, Ellwangen	06/2011
Caliqua, Basel, Switzerland	06/2011
Technical University of Freiberg	06/2011
Fichtner IT Consulting, Stuttgart	05/2011, 06/2011, 08/2011
Salzgitter Flachstahl, Salzgitter	05/2011
Helbling Beratung & Bauplanung, Zurich, Switzerland	05/2011
INEOS, Cologne	04/2011
Enseleit Consulting Engineers, Siebigerode	04/2011
Witt Consulting Engineers, Stade	03/2011

Helbling, Zurich, Switzerland	03/2011
MAN Diesel, Copenhagen, Denmark	03/2011
AGO, Kulmbach	03/2011
University of Duisburg	03/2011, 06/2011
CCP, Marburg	03/2011
BASF, Ludwigshafen	02/2011
ALSTOM Power, Baden, Switzerland	02/2011
Universität der Bundeswehr, Munich	02/2011
Calorifer, Elgg, Switzerland	01/2011
STRABAG, Vienna, Austria	01/2011
TUEV Sued, Munich	01/2011
ILK Dresden	01/2011
Technical University of Dresden	01/2011, 05/2011, 06/2011, 08/2011

2010

Umweltinstitut Neumarkt	12/2010
YIT Austria, Vienna, Austria	12/2010
MCI Innsbruck, Austria	12/2010
University of Stuttgart	12/2010
HS Cooler, Wittenburg	12/2010
Visteon, Novi Jicin, Czech Republic	12/2010
CompuWave, Brunntal	12/2010
Stadtwerke Leipzig	12/2010
MCI Innsbruck, Austria	12/2010
EVONIK Energy Services, Zwingenberg	12/2010
Caliqua, Basel, Switzerland	11/2010
Shanghai New Energy Resources Science & Technology, China	11/2010
Energieversorgung Halle	11/2010
Hochschule für Technik Stuttgart, University of Applied Sciences	11/2010
Steinmueller, Berlin	11/2010
Amberg-Weiden University of Applied Sciences	11/2010
AREVA NP, Erlangen	10/2010
MAN Diesel, Augsburg	10/2010
KRONES, Neutraubling	10/2010
Vaillant, Remscheid	10/2010

PC Ware, Leipzig	10/2010
Schubert Consulting Engineers, Weißenberg	10/2010
Fraunhofer Institut UMSICHT, Oberhausen	10/2010
Behringer Consulting Engineers, Tagmersheim	09/2010
Saacke, Bremen	09/2010
WEBASTO, Neubrandenburg	09/2010
Concordia University, Montreal, Canada	09/2010
Compañía Eléctrica de Sochagota, Bogota, Colombia	08/2010
Hannover University of Applied Sciences	08/2010
ERGION, Mannheim	07/2010
Fichtner IT Consulting, Stuttgart	07/2010
TF Design, Matieland, South Africa	07/2010
MCE, Berlin	07/2010, 12/2010
IPM, Zittau/Goerlitz University of Applied Sciences	06/2010
TUEV Sued, Dresden	06/2010
RWE IT, Essen	06/2010
Glen Dimplex, Kulmbach	05/2010, 07/2010 10/2010
Hot Rock, Karlsruhe	05/2010
Darmstadt University of Applied Sciences	05/2010
Voith, Heidenheim	04/2010
CombTec, Zittau	04/2010
University of Glasgow, Great Britain	04/2010
Universitaet der Bundeswehr, Munich	04/2010
Technical University of Hamburg-Harburg	04/2010
Vattenfall Europe, Berlin	04/2010
HUBER Consulting Engineers, Berching	04/2010
VER, Dresden	04/2010
CCP, Marburg	03/2010
Offenburg University of Applied Sciences	03/2010
Technical University of Berlin	03/2010
NIST Boulder CO, USA	03/2010
Technical University of Dresden	02/2010
Siemens Energy, Nuremberg	02/2010
Augsburg University of Applied Sciences	02/2010

ALSTOM Power, Baden, Switzerland	02/2010, 05/2010
MIT Massachusetts Institute of Technology Cambridge MA, USA	02/2010
Wieland Werke, Ulm	01/2010
Siemens Energy, Goerlitz	01/2010, 12/2010
Technical University of Freiberg	01/2010
ILK, Dresden	01/2010, 12/2010
Fischer-Uhrig Consulting Engineers, Berlin	01/2010

2009

ALSTOM Power, Baden, Schweiz	01/2009, 03/2009, 05/2009
Nordostschweizerische Kraftwerke AG, Doettingen, Switzerland	02/2009
RWE, Neurath	02/2009
Brandenburg University of Technology, Cottbus	02/2009
Hamburg University of Applied Sciences	02/2009
Kehrein, Moers	03/2009
EPP Software, Marburg	03/2009
Bernd Münstermann, Telgte	03/2009
Suedzucker, Zeitz	03/2009
CPP, Marburg	03/2009
Gelsenkirchen University of Applied Sciences	04/2009
Regensburg University of Applied Sciences	05/2009
Gatley & Associates, Atlanta, USA	05/2009
BOSCH, Stuttgart	06/2009, 07/2009
Dr. Nickolay, Consulting Engineers, Gommersheim	06/2009
Ferrostal Power, Saarlouis	06/2009
BHR Bilfinger, Essen	06/2009
Intraserv, Wiesbaden	06/2009
Lausitz University of Applied Sciences, Senftenberg	06/2009
Nuernberg University of Applied Sciences	06/2009
Technical University of Berlin	06/2009
Fraunhofer Institut UMSICHT, Oberhausen	07/2009
Bischoff, Aurich	07/2009
Fichtner IT Consulting, Stuttgart	07/2009
Techsoft, Linz, Austria	08/2009
DLR, Stuttgart	08/2009

Wienstrom, Vienna, Austria	08/2009
RWTH Aachen University	09/2009
Vattenfall, Hamburg	10/2009
AIC, Chemnitz	10/2009
Midiplan, Bietigheim-Bissingen	11/2009
Institute of Air Handling and Refrigeration ILK, Dresden	11/2009
FZD, Rossendorf	11/2009
Techgroup, Ratingen	11/2009
Robert Sack, Heidelberg	11/2009
EC, Heidelberg	11/2009
MCI, Innsbruck, Austria	12/2009
Saacke, Bremen	12/2009
ENERKO, Aldenhoven	12/2009

2008

Pink, Langenwang	01/2008
Fischer-Uhrig, Berlin	01/2008
University of Karlsruhe	01/2008
MAAG, Kuesnacht, Switzerland	02/2008
M&M Turbine Technology, Bielefeld	02/2008
Lentjes, Ratingen	03/2008
Siemens Power Generation, Goerlitz	04/2008
Evonik, Zwingenberg (general EBSILON program license)	04/2008
WEBASTO, Neubrandenburg	04/2008
CFC Solutions, Munich	04/2008
RWE IT, Essen	04/2008
Rerum Cognitio, Zwickau	04/2008, 05/2008
ARUP, Berlin	05/2008
Research Center, Karlsruhe	07/2008
AWECO, Neukirch	07/2008
Technical University of Dresden, Professorship of Building Services	07/2008
Technical University of Cottbus, Chair in Power Plant Engineering	07/2008, 10/2008
Ingersoll-Rand, Unicov, Czech Republic	08/2008
Technip Benelux BV, Zoetermeer, Netherlands	08/2008
Fennovoima Oy, Helsinki, Finland	08/2008

Fichtner Consulting & IT, Stuttgart	09/2008
PEU, Espenhain	09/2008
Poyry, Dresden	09/2008
WINGAS, Kassel	09/2008
TUEV Sued, Dresden	10/2008
Technical University of Dresden, Professorship of Thermic Energy Machines and Plants	10/2008, 11/2008
AWTEC, Zurich, Switzerland	11/2008
Siemens Power Generation, Erlangen	12/2008

2007

Audi, Ingolstadt	02/2007
ANO Abfallbehandlung Nord, Bremen	02/2007
TUEV NORD SysTec, Hamburg	02/2007
VER, Dresden	02/2007
Technical University of Dresden, Chair in Jet Propulsion Systems	02/2007
Redacom, Nidau, Switzerland	02/2007
Universität der Bundeswehr, Munich	02/2007
Maxxtec, Sinsheim	03/2007
University of Rostock, Chair in Technical Thermodynamics	03/2007
AGO, Kulmbach	03/2007
University of Stuttgart, Chair in Aviation Propulsions	03/2007
Siemens Power Generation, Duisburg	03/2007
ENTHAL Haustechnik, Rees	05/2007
AWECO, Neukirch	05/2007
ALSTOM, Rugby, Great Britain	06/2007
SAAS, Possendorf	06/2007
Grenzebach BSH, Bad Hersfeld	06/2007
Reichel Engineering, Haan	06/2007
Technical University of Cottbus, Chair in Power Plant Engineering	06/2007
Voith Paper Air Systems, Bayreuth	06/2007
Egger Holzwerkstoffe, Wismar	06/2007
Tissue Europe Technologie, Mannheim	06/2007
Dometic, Siegen	07/2007
RWTH Aachen University, Institute for Electrophysics	09/2007
National Energy Technology Laboratory, Pittsburg, USA	10/2007

Energieversorgung Halle	10/2007
AL-KO, Jettingen	10/2007
Grenzebach BSH, Bad Hersfeld	10/2007
Wiesbaden University of Applied Sciences, Department of Engineering Sciences	10/2007
Endress+Hauser Messtechnik, Hannover	11/2007
Munich University of Applied Sciences, Department of Mechanical Engineering	11/2007
Rerum Cognitio, Zwickau	12/2007
Siemens Power Generation, Erlangen	11/2007
University of Rostock, Chair in Technical Thermodynamics	11/2007, 12/2007

2006

STORA ENSO Sachsen, Eilenburg	01/2006
Technical University of Munich, Chair in Energy Systems	01/2006
NUTEC Engineering, Bisikon, Switzerland	01/2006, 04/2006
Conwel eco, Bochov, Czech Republic	01/2006
Offenburg University of Applied Sciences	01/2006
KOCH Transporttechnik, Wadgassen	01/2006
BEG Bremerhavener Entsorgungsgesellschaft	02/2006
Deggendorf University of Applied Sciences, Department of Mechanical Engineering and Mechatronics	02/2006
University of Stuttgart, Department of Thermal Fluid Flow Engines	02/2006
Technical University of Munich, Chair in Apparatus and Plant Engineering	02/2006
Energietechnik Leipzig (company license),	02/2006
Siemens Power Generation, Erlangen	02/2006, 03/2006
RWE Power, Essen	03/2006
WAETAS, Pobershau	04/2006
Siemens Power Generation, Goerlitz	04/2006
Technical University of Braunschweig, Department of Thermodynamics	04/2006
EnviCon & Plant Engineering, Nuremberg	04/2006
Brassel Engineering, Dresden	05/2006
University of Halle-Merseburg, Department of USET Merseburg incorporated society	05/2006
Technical University of Dresden, Professorship of Thermic Energy Machines and Plants	05/2006

Fichtner Consulting & IT Stuttgart (company licenses and distribution)	05/2006
Suedzucker, Ochsenfurt	06/2006
M&M Turbine Technology, Bielefeld	06/2006
Feistel Engineering, Volkach	07/2006
ThyssenKrupp Marine Systems, Kiel	07/2006
Caliqua, Basel, Switzerland (company license)	09/2006
Atlas-Stord, Rodovre, Denmark	09/2006
Konstanz University of Applied Sciences, Course of Studies Construction and Development	10/2006
Siemens Power Generation, Duisburg	10/2006
Hannover University of Applied Sciences, Department of Mechanical Engineering	10/2006
Siemens Power Generation, Berlin	11/2006
Zikesch Armaturentechnik, Essen	11/2006
Wismar University of Applied Sciences, Seafaring Department	11/2006
BASF, Schwarzheide	12/2006
Enertech Energie und Technik, Radebeul	12/2006

2005

TUEV Nord, Hannover	01/2005
J.H.K Plant Engineering and Service, Bremerhaven	01/2005
Electrowatt-EKONO, Zurich, Switzerland	01/2005
FCIT, Stuttgart	01/2005
Energietechnik Leipzig (company license)	02/2005, 04/2005, 07/2005
eta Energieberatung, Pfaffenhofen	02/2005
FZR Forschungszentrum, Rossendorf/Dresden	04/2005
University of Saarbruecken	04/2005
Technical University of Dresden Professorship of Thermic Energy Machines and Plants	04/2005
Grenzebach BSH, Bad Hersfeld	04/2005
TUEV Nord, Hamburg	04/2005
Technical University of Dresden, Waste Management	05/2005
Siemens Power Generation, Goerlitz	05/2005
Duesseldorf University of Applied Sciences, Department of Mechanical Engineering and Process Engineering	05/2005

Redacom, Nidau, Switzerland	06/2005
Dumas Verfahrenstechnik, Hofheim	06/2005
Alensys Engineering, Erkner	07/2005
Stadtwerke Leipzig	07/2005
SaarEnergie, Saarbruecken	07/2005
ALSTOM ITC, Rugby, Great Britain	08/2005
Technical University of Cottbus, Chair in Power Plant Engineering	08/2005
Vattenfall Europe, Berlin (group license)	08/2005
Technical University of Berlin	10/2005
Basel University of Applied Sciences, Department of Mechanical Engineering, Switzerland	10/2005
Midiplan, Bietigheim-Bissingen	11/2005
Technical University of Freiberg, Chair in Hydrogeology	11/2005
STORA ENSO Sachsen, Eilenburg	12/2005
Energieversorgung Halle (company license)	12/2005
KEMA IEV, Dresden	12/2005

2004

Vattenfall Europe (group license)	01/2004
TUEV Nord, Hamburg	01/2004
University of Stuttgart, Institute of Thermodynamics and Heat Engineering	02/2004
MAN B&W Diesel A/S, Copenhagen, Denmark	02/2004
Siemens AG Power Generation, Erlangen	02/2004
Ulm University of Applied Sciences	03/2004
Visteon, Kerpen	03/2004, 10/2004
Technical University of Dresden, Professorship of Thermic Energy Machines and Plants	04/2004
Rerum Cognitio, Zwickau	04/2004
University of Saarbruecken	04/2004
Grenzebach BSH, Bad Hersfeld	04/2004
SOFBID Zwingenberg (general EBSILON program license)	04/2004
EnBW Energy Solutions, Stuttgart	05/2004
HEW-Kraftwerk, Tiefstack	06/2004
h s energieanlagen, Freising	07/2004
FCIT, Stuttgart	08/2004
Physikalisch Technische Bundesanstalt (PTB), Braunschweig	08/2004
Mainova Frankfurt	08/2004

Rietschle Energieplaner, Winterthur, Switzerland	08/2004
MAN Turbo Machines, Oberhausen	09/2004
TUEV Sued, Dresden	10/2004
STEAG Kraftwerk, Herne	10/2004, 12/2004
University of Weimar	10/2004
energeticals (e-concept), Munich	11/2004
SorTech, Halle	11/2004
Enertech EUT, Radebeul (company license)	11/2004
Munich University of Applied Sciences	12/2004
STORA ENSO Sachsen, Eilenburg	12/2004
Technical University of Cottbus, Chair in Power Plant Engineering	12/2004
Freudenberg Service, Weinheim	12/2004

2003

Paper Factory, Utzenstorf, Switzerland	01/2003
MAB Plant Engineering, Vienna, Austria	01/2003
Wulff Energy Systems, Husum	01/2003
Technip Benelux BV, Zoetermeer, Netherlands	01/2003
ALSTOM Power, Baden, Switzerland	01/2003, 07/2003
VER, Dresden	02/2003
Rietschle Energieplaner, Winterthur, Switzerland	02/2003
DLR, Leupholdhausen	04/2003
Emden University of Applied Sciences, Department of Technology	05/2003
Pettersson+Ahrends, Ober-Moerlen	05/2003
SOFBID ,Zwingenberg (general EBSILON program license)	05/2003
Ingenieurbuero Ostendorf, Gummersbach	05/2003
TUEV Nord, Hamburg	06/2003
Muenstermann GmbH, Telgte-Westbevern	06/2003
University of Cali, Colombia	07/2003
Atlas-Stord, Rodovre, Denmark	08/2003
ENERKO, Aldenhoven	08/2003
STEAG RKB, Leuna	08/2003
eta Energieberatung, Pfaffenhofen	08/2003
exergie, Dresden	09/2003
AWTEC, Zurich, Switzerland	09/2003
Energie, Timelkam, Austria	09/2003

Electrowatt-EKONO, Zurich, Switzerland	09/2003
LG, Annaberg-Buchholz	10/2003
FZR Forschungszentrum, Rossendorf/Dresden	10/2003
EnviCon & Plant Engineering, Nuremberg	11/2003
Visteon, Kerpen	11/2003
VEO Vulkan Energiewirtschaft Oderbruecke, Eisenhuettenstadt	11/2003
Stadtwerke Hannover	11/2003
SaarEnergie, Saarbruecken	11/2003
Fraunhofer-Gesellschaft, Munich	12/2003
Erfurt University of Applied Sciences, Department of Supply Engineering	12/2003
SorTech, Freiburg	12/2003
Mainova, Frankfurt	12/2003
Energieversorgung Halle	12/2003

2002

Hamilton Medical AG, Rhaezuens, Switzerland	01/2002
Bochum University of Applied Sciences, Department of Thermo- and Fluid Dynamics	01/2002
SAAS, Possendorf/Dresden	02/2002
Siemens, Karlsruhe (general license for the WinIS information system)	02/2002
FZR Forschungszentrum, Rossendorf/Dresden	03/2002
CompAir, Simmern	03/2002
GKS Gemeinschaftskraftwerk, Schweinfurt	04/2002
ALSTOM Power Baden, Switzerland (group licenses)	05/2002
InfraServ, Gendorf	05/2002
SoftSolutions, Muehlhausen (company license)	05/2002
DREWAG, Dresden (company license)	05/2002
SOFBID, Zwingenberg (general EBSILON program license)	06/2002
Kleemann Engineering, Dresden	06/2002
Caliqua, Basel, Switzerland (company license)	07/2002
PCK Raffinerie, Schwedt (group license)	07/2002
Fischer-Uhrig Engineering, Berlin	08/2002
Fichtner Consulting & IT, Stuttgart (company licenses and distribution)	08/2002
Stadtwerke Duisburg	08/2002

Stadtwerke Hannover	09/2002
Siemens Power Generation, Goerlitz	10/2002
Energieversorgung Halle (company license)	10/2002
Bayer, Leverkusen	11/2002
Dillinger Huette, Dillingen	11/2002
G.U.N.T. Geraetebau, Barsbuettel (general license and training test benches)	12/2002
VEAG, Berlin (group license)	12/2002

2001

ALSTOM Power, Baden, Switzerland	01/2001, 06/2001, 12/2001
KW2 B. V., Amersfoot, Netherlands	01/2001, 11/2001
Eco Design, Saitamaken, Japan	01/2001
M&M Turbine Technology, Bielefeld	01/2001, 09/2001
MVV Energie, Mannheim	02/2001
Technical University of Dresden, Department of Power Machinery and Plants	02/2001
PREUSSAG NOELL, Wuerzburg	03/2001
Fichtner Consulting & IT Stuttgart (company licenses and distribution)	04/2001
Muenstermann GmbH, Telgte-Westbevern	05/2001
SaarEnergie, Saarbruecken	05/2001
Siemens, Karlsruhe (general license for the WinIS information system)	08/2001
Neusiedler AG, Ulmerfeld, Austria	09/2001
h s energieanlagen, Freising	09/2001
Electrowatt-EKONO, Zurich, Switzerland	09/2001
IPM Zittau/Goerlitz University of Applied Sciences (general license)	10/2001
eta Energieberatung, Pfaffenhofen	11/2001
ALSTOM Power Baden, Switzerland	12/2001
VEAG, Berlin (group license)	12/2001

2000

SOFBID, Zwingenberg (general EBSILON program license)	01/2000
AG KKK - PGW Turbo, Leipzig	01/2000
PREUSSAG NOELL, Wuerzburg	01/2000
M&M Turbine Technology, Bielefeld	01/2000

IBR Engineering Reis, Nittendorf-Undorf	02/2000
GK, Hannover	03/2000
KRUPP-UHDE, Dortmund (company license)	03/2000
UMAG W. UDE, Husum	03/2000
VEAG, Berlin (group license)	03/2000
Thinius Engineering, Erkrath	04/2000
SaarEnergie, Saarbruecken	05/2000, 08/2000
DVO Data Processing Service, Oberhausen	05/2000
RWTH Aachen University	06/2000
VAUP Process Automation, Landau	08/2000
Knuerr-Lommatec, Lommatzsch	09/2000
AVACON, Helmstedt	10/2000
Compania Electrica, Bogota, Colombia	10/2000
G.U.N.T. Geraetebau, Barsbuettel (general license for training test benches)	11/2000
Steinhaus Informationssysteme, Datteln (general license for process data software)	12/2000

1999

Bayernwerk, Munich	01/1999
DREWAG, Dresden (company license)	02/1999
KEMA IEV, Dresden	03/1999
Regensburg University of Applied Sciences	04/1999
Fichtner Consulting & IT, Stuttgart (company licenses and distribution)	07/1999
Technical University of Cottbus, Chair in Power Plant Engineering	07/1999
Technical University of Graz, Department of Thermal Engineering, Austria	11/1999
Ostendorf Engineering, Gummersbach	12/1999

1998

Technical University of Cottbus, Chair in Power Plant Engineering	05/1998
Fichtner Consulting & IT (CADIS information systems) Stuttgart (general KPRO program license)	05/1998
M&M Turbine Technology Bielefeld	06/1998
B+H Software Engineering Stuttgart	08/1998
Alfa Engineering, Switzerland	09/1998
VEAG Berlin (group license)	09/1998
NUTEC Engineering, Bisikon, Switzerland	10/1998

SCA Hygiene Products, Munich	10/1998
RWE Energie, Neurath	10/1998
Wilhelmshaven University of Applied Sciences	10/1998
BASF, Ludwigshafen (group license)	11/1998
Energieversorgung, Offenbach	11/1998

1997

Gerb, Dresden	06/1997
Siemens Power Generation, Goerlitz	07/1997