FFI-NOTAT

Eksternnotat 21/00238

LybinTCPserver 7.0.3

Interface description

Author
Elin Bøhler
Prosjektnummer 549701
1 February 2021

Approvers

Connie Elise Solberg, *Research Manager*; Trygve Sparr, *Research Director*.
The document is electronically approved and therefore has no handwritten signature.

Keywords
Undervannsakustikk, Sonar, LYBIN, Programvare, Grensesnitt

Summary
LYBIN is a robust, user friendly and fast acoustic ray-trace simulator. A broad set of parameters are used to accurately calculate the probability of detecting objects in a given area under water with the use of sonar technology. LYBIN can be used both with a graphical user interface and as a stand-alone calculation kernel. The stand-alone calculation kernel is available in two different implementations; LybinCom and LybinTCPserver. This FFI note describes the interface of LybinTCPserver 7.0.3.

Contents

1 Introduction **3**

2 LYBIN model data **4**

2.1 Environment 10

2.2 Platform 28

3 Initiate calculation **32**

4 Calculation results **32**

4.1 Functions returning calculation results 32

4.2 Impulseresponse point 34

4.3 Traveltime point 36

4.4 Visual raytrace point 36

References **37**

1 Introduction

LYBIN [1], [2] is a well established and frequently used sonar prediction tool owned by the Norwegian Defence Materiel Agency (NDMA) and FFI. It is in operative use by the Norwegian Navy and in a number of other nations, and has been modified and improved for this purpose for more than 30 years. FFI has been responsible for testing, evaluation and development of LYBIN since 2000 and has been responsible for commercial sale and support since 2009.

LYBIN is a robust, user friendly and fast acoustic ray-trace simulator. A broad set of parameters are used to accurately calculate the probability of detecting objects in a given area under water with the use of sonar technology. As this probability changes with environmental properties, LYBIN rapidly calculates the sonar coverage.

Several thousand acoustic rays are simulated traversing the water volume. Upon hitting the sea surface and sea bed, the rays are reflected and exposed to loss mechanisms. Losses in the water volume itself, due to thermal absorption are accounted for. LYBIN estimates the probability of detection for a given target, based on target echo strength, the calculated transmission loss, reverberation and noise. Both active and passive sonar systems can be simulated.

LYBIN can be used both with a graphical user interface [3] and as a stand-alone calculation kernel. This duality enables LYBIN to interact with other applications, such as mathematical models, web services, geographic information systems, and more. The software is integrated in combat system software, tactical decision aids and tactical trainers. LYBIN has become an important tool in both planning and evaluation of maritime operations [4],[5].

The stand-alone calculation kernel is available in two different implementations; LybinCom and LybinTCPserver. LybinCom [6] is implemented as a Microsoft COM [7] module for the Windows platform. LybinTCPserver is based on Apache Thrift [8], using TCP/IP remote procedure calls. LybinTCPserver can be built for both Windows and Linux platforms and used from multiple programming languages.

This FFI note describes the interface of LybinTCPserver 7.0.3. The three following chapters describe the separate parts of the interface. Chapter 2 gives a description of all the input parameters that can be used in the simulations. Chapter 3 gives a description of how to initiate a sonar performance calculation and Chapter 4 gives a description of all the calculation results available from the calculation.

2 LYBIN model data

The LybinModelData class contains all the parameters to be used in a simulation, the environment, the platform and all the parameters controlling the acoustic calculations: the resolution of the calculation, what type of calculation to be performed, and so on. All the parameters in LybinModelData are listed in *Table 2.1*.

Parameter	Type	Default value	Unit
DepthCells *Number of depth cells in the calculation output.* *DepthCell is read only, so the function SetDepthScaleAndDepthCells must be used to set this parameter directly.*	Integer	50	
DepthCellSize *Size of the depth cells in the calculation output.* *DepthCellSize is read only, so the functions SetDepthCellSizeAndDepthSteps or SetDepthScaleAndDepthCellSize must be used to set this parameter directly.*	Double	6	m
DepthScale *Maximum depth in the calculation.*	Double	300	m
DepthSteps *Number of depth steps to be used during the calculation.* *DepthSteps is read only, so the functions SetDepthCellSizeAndDepthSteps or SetDepthScaleAndDepthCellSteps must be used to set this parameter directly.*	Integer	1000	
DepthStepSize *Size of the depth steps to be used during the calculation.* *This parameter is read only, and is derived by other depth calculation parameters.*	Double	0.3	m

Parameter	Type	Default value	Unit
Environment *All the environmental data to be used un the calculation.*	Environment		
ImpulseResponseCalculation *Switch to control whether to calculate impulse response or not.* *False: Do not calculate impulse response.* *True: Calculate impulse response.*	Boolean	false	
ImpulseResponseDepth *The depth that the impulse response will be calculated from.*	Double	0	m
ImpulseResponseWindowHeight *The higth of the windowthat the impulse response will be calculated from.*	Double	80	m
MaxBorderHits *Maximum number of boundary hits (sea or bottom) allowed before a ray is terminated.*	Integer	5000	
NoiseCalculation *Switch to control whether to calculate the noise or not.* *False: Do not calculate noise.* *True: Do calculate noise.*	Boolean	true	
PassiveCalculation *Switch to control whether to perform calculations for active or passive sonar.* *False: Calculate for active sonar.* *True: Calculate for passive sonar.*	Boolean	false	
Platform *All the platform data to be used in the calculation.*	Platform		

Parameter	Type	Default value	Unit
RangeCells *Number of range cells in the calculation output.* *RangeCells is read only, so the function SetRangeScaleAndRangeCells must be used to set this parameter directly.*	Integer	50	
RangeCellSize *Size of the range cells in the calculation output.* *RangeCellSize is read only, so the functions SetRangeCellSizeAndRangeSteps or SetRangeScaleAndRangeCellSize must be used to set this parameter directly.*	Double	200	m
RangeScale *Maximum range in the calculat*ion.	Double	10000	m
RangeSteps *Number of range steps to be used during the calculation.* *RangeSteps is read only, so the functions SetRangeCellSizeAndRangeSteps or SetRangeScaleAndRangeCellSteps must be used to set this parameter directly.*	Integer	500	
RangeStepSize *Size of the range steps to be used during the calculation.* *RangeStepSize is read only, and it is derived by other range calculation parameters.*	Double	20	m
SignalExcessConstant *Parameter affecting the relation between signal excess and probability of detection.*	Double	3	
TerminationIntensity *Each ray is terminated when its intensity falls below this value.*	Double	1E-16	

Parameter	Type	Default value	Unit
TravelTimeAngleRes *The distance in degrees between the start angles of the rays to be used in the travel time calculation.*	Double	1	Deg
DoTravelTimeCalculation *Switch to control whether to calculate travel time or not.* *False: Do not calculate travel time.* *True: Calculate travel time.*	Boolean	false	
TRLRays *Number of rays to be used in the transmission loss calculation.*	Integer	1000	
TypeOfRevNoiseCalculation *Enumerator used to control how the calculation of reverberation is performed:* *0: Calculate bottom reverberation from bottom types* *1: Calculate bottom reverberation from back scatter values* *2: Use measured reverberation and noise data* *3: Use Lamberts law to calculate bottom reverberation*	Integer	0	
UseMeasuredBottomLoss *Tells the model how to calculate bottom loss. If UseRayleighBottomLoss = true, it will overrule UseMeasuredBottomLoss.* *False: Use bottom types to calculate bottom loss* *True: Use measured or supplied bottom loss values*	Boolean	false	

Parameter	Type	Default value	Unit
UseMeasuredHorizontalBeamWidth *Tells whether to use the input parameter BeamWidthHorizontal instead of calculating the horizontal beam.* *False:* *Use directivity index and vertical beam width to calculate horizontal beam width* *True:* *Use measured or supplied horizontal beam width*	Boolean	false	
UseMeasuredPassiveProcessingGain *Tells whether to use the input parameter PassiveProcessingGain instead of calculating the passive processing gain.* *False:* *Calculate passive processing gain based on type of sonar, beamwidth and integration time* *True:* *Use measured or supplied passive processing gain*	Boolean	false	
UseMeasuredSurfaceBackScatter *Tells the model how to calculate surface back scatter.* *False:* *Use wind speed to calculate surface back scatter* *True:* *Use measured or supplied surface back scatter*	Boolean	false	
UseMeasuredSurfaceLoss *Tells the model how to calculate surface loss.* *False:* *Use wind speed to calculate surface loss* *True:* *Use measured or supplied surface loss values*	Boolean	false	
UseSurfaceReflectionAngles *Tells the model how to calculate surface reflection angles.* *False:* *Ray tracing algorithms to calculate surface reflection angles* *True:* *Use measured or supplied surface reflection angles*	Boolean	false	

Parameter	Type	Default value	Unit
UseMeasuredTargetStrength *Tells the model* *False:* *Use the single parameter TargetStrengthto calculatetarget strength* *True:* *Use measured or supplied from a target strength file*	Boolean	false	
UseRayleighBottomLoss *Tells the model how to calculate bottom loss. If UseRayleighBottomLoss = true, it will overrule UseMeasuredBottomLoss.* *False:* *Use Rayleigh bottom loss* *True:* *Do not use Rayleigh bottom loss.*	Boolean	false	
UseWaveHeight *Tells the model to use wave height instead of wind speed.* *False:* *Use wind speed.* *True:* *Use wave height.*	Boolean	false	
VisualRayTraceCalculation *Switch to control whether to calculate a ray trace plot for visualisation or not.* *False:* *Do not calculate ray trace for visualisation.* *True:* *Calculate ray trace for visualisation.*	Boolean	false	
VisualBottomHits *Number of bottom hits alloved in the visual ray trace plot.*	Integer	1	
VisualNumRays *Number of rays in the visual ray trace plot.*	Integer	50	
VisualSurfaceHits *Number of surface hits alloved in the visual ray trace plot.*	Integer	2	

Table 2.1 Parameters in the LybinModelData class.

TypeOfRevNoiseCalculation, UseMeasuredBottomLoss, UseMeasuredHorizontalBeamWidth, UseMeasuredPassiveProsessingGain, UseMeasuredSurfaceBackScatter, UseMeasuredSurfaceLoss, UseMeasuredSurfaceReflectionAngles, UseMeasuredTargetStrength and UseRayleighBottomLoss can make LybinTCPserver use certain datasets instead of predefined default values. In order to follow these demands, the spesified datasets must be sent into LybinTCPserver. If LybinTCPserver cannot find these datasets, the switches will be set back to default values.

The relation between cells and steps is by default so that the number of range steps is 10 times the number of range cells and the number of depth steps is 20 times the number of depth cells. To avoid to large steps, there is a maximum range step size of 50 meters and a maximum depth step size of 5 meters. If the maximum size is exceeded, additional steps are added.

2.1 Environment

The environment class contains all the environmental data as listed in *Table 2.2.*

LybinTCPserver is able to handle range dependent environments. In LybinTCPserver, range dependent environmental data are specified for certain range intervals from the sonar.

When the environmental properties are entered for a discrete set of locations (ranges), LybinTCPserver will create values at intermediate ranges using interpolation. If no environmental descriptions are given at zero range, LybinTCPserver will substitute the data for the nearest range available, likewise, if data at maximum range are missing.

We call these datasets, with start and stop related to a value (or sets of values), for a range dependent object. A range dependent object can contain one or more values with their range of validity. The structure of range dependent objects, with start and stop range is shown in *Figure 2.1* . The maximum number of range dependent values are only limited by the given calculation accuracy.

The start and stop functionality provides great flexibility in defining the environmental range dependent properties. By setting start and stop to the same range, the values will be considered to belong to a point in space, and LybinTCPserver will use interpolation to produce data for intermediate ranges points. The start and stop functionality might be utilized to illustrate meteorological or oceanographic fronts, entering ranges with finite ranges of validity to each side of the front, and separating the sets by any small distance, across which the conditions will change as abruptly as the user intends. In between these two extreme choices all combination of these are possible to use.

The BottomProfile and the ReverberationAndNoiseMeasurements do not have the start-stop functionality. These datasets are not likely to have constant values over range. Both BottomProfile and the ReverberationAndNoiseMeasurements are to be inserted into LybinTCPserver as single values with corresponding range. The number of data points in each dataset is optional.

Parameter	Type
BottomBackScatter	List<StartStopDoubleList>
BottomLoss	List<StartStopDoubleList>
BottomProfile	List<BottomProfileSample>
BottomType	List<StartStopSampleDouble>
LambertsCoefficient	List<StartStopSampleDouble>
Ocean	Ocean
RayleighBottomLoss	RayleighBottomLoss
ReverberationAndNoise	List<ReverberationAndNoiseSample>
SoundSpeed	List<SoundSpeedProfile>
SurfaceBackScatter	List<StartStopDoubleList>
SurfaceLoss	List<StartStopDoubleList>
SurfaceReflectionAngle	List<StartStopSampleDouble>
TargetStrength	List<StartStopDoubleList>
VolumeBackScatter	List<VolumeBackScatterProfile>
WaveHeight	List<StartStopSampleDouble>
WindSpeed	List<StartStopSampleDouble>

Table 2.2 The environment class holds all the environment data.

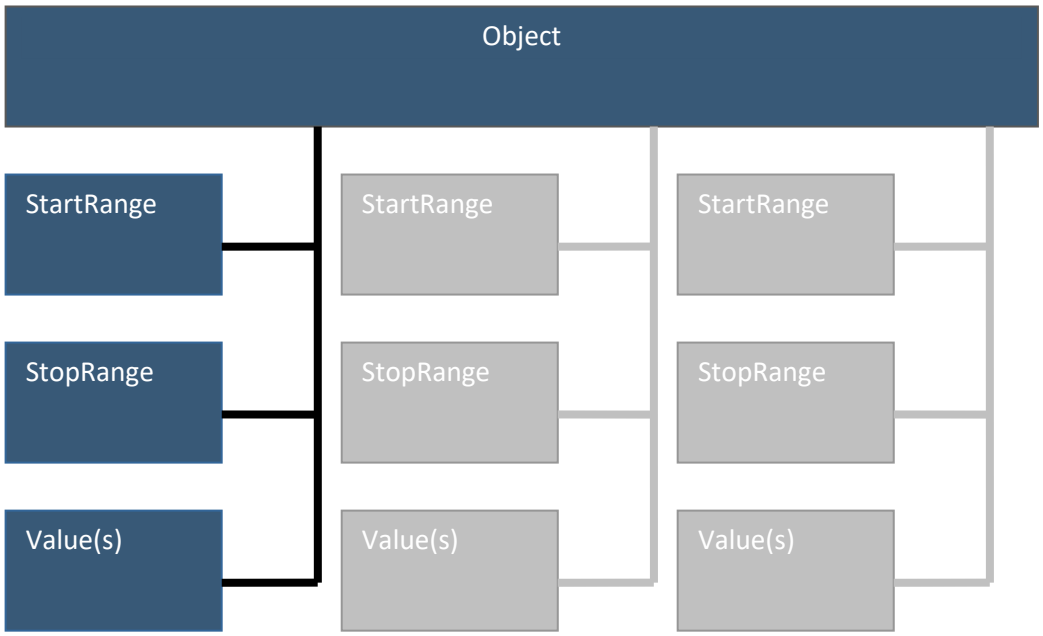

Figure 2.1 Schematic description of a range dependent object with start and stop parameters.

2.1.1 Bottom back scatter

Bottom back scatter is the fraction of energy that is scattered back towards to the receiver when a ray hits the sea bottom. The bottom back scattering is generally a function of bottom type, grazing angle and frequency. A dataset representing bottom back scattering coefficients is entered into LybinTCPserver in tabular form, giving backscattering coefficients (in dB) for a set of grazing angles. Based on the tabulated values, LybinTCPserver interpolates between tabulated values to create backscattering coefficients for equidistantly spaced grazing angles. The back scattering coefficients are given as dB per square meter.

Bottom back scatter is one of four possible options to calculate bottom reverberation. LybinTCPserver will only use the bottom back scatter values given if the TypeOfRevNoiseCalculation parameter in LybinModelData class is set to 1 (Calculate bottom reverberation from back scatter values).

A start stop double list containing range dependent bottom back scatter values can be added to the BottomBackScatter class as listed in *Table 2.3* and *Table 2.4*.

Class	Parameter	Type	Default value	Unit
DoubleSample	**Data**	Double	0	dB
	Key	Double	0	deg
StartStopDoubleList	**Start**	Integer	0	m
	Stop	Integer	0	m
	Samples	List<DoubleSample>		

Table 2.3 Bottom back scatter start stop list contains start, stop and bottom back scatter double samples.

Function	Type
Add(StartStopDoubleList bottomBackScatterTable) *Add a range dependent bottom back scatter table.*	*Void*

Table 2.4 Add a range dependent bottom back scatter table to the BottomBackScatter class.

2.1.2 Bottom loss

Bottom loss is the fraction of energy that is lost after the sound has been reflected from the ocean bottom, usually expressed in dB. The bottom loss is also referred to as *forward scattering* in underwater acoustic terminology. Bottom loss is generally a function of bottom type, gracing angle and frequency. A dataset representing bottom loss is entered into LybinTCPserver in tabular form, giving bottom loss (in dB) for a set of grazing angles. Based on the tabulated values, LybinTCPserver interpolates between tabulated values to create loss values for equidistantly spaced grazing angles.

The parameter UseMeasuredBottomLoss tells LybinTCPserver to use BottomLossTable instead of calculating the bottom loss. If UseRayleighBottomLoss is set to true, UseMeasuredBottomLoss will be ignored. UseRayleighBottomLoss must always be set to false and UseMeasuredBottomLoss to true if one wants to use predefined bottom loss values in LybinTCPserver. Both these parameters can be found in the LybinModelData class.

A start stop double list containing range dependent bottom loss values can be added to the BottomLoss class as listed in *Table 2.6*.

Class	Parameter	Type	Default value	Unit
DoubleSample	**Data**	Double	0	dB
	Key	Double	0	deg
StartStopDoubleList	**Start**	Integer	0	m
	Stop	Integer	0	m
	Samples	List<DoubleSample>		

Table 2.5 Bottom loss start stop list contains start, stop and bottom loss double samples.

Function	Type
Add(StartStopDoubleList bottomLossTable) *Add a range dependent bottom loss table.*	*Void*

Table 2.6 Add a range dependent bottom loss table to the BottomLoss class.

2.1.3 Bottom profile

The BottomProfile consist of bottom profile samples containing range and depth values as listed in *Table 2.7*. The samples are added to the bottom profile using the Add function as listed in *Table 2.8.*

Class	Parameter	Type	Default value	Unit
BottomProfileSample	**Depth**	Double	200	m
	Range	Double	0	m

Table 2.7 The BottomProfileSample contains range and depth.

Function	Type
Add(BottomProfileSample bottomProfileSample) *Add a bottom profile sample containing range and depth..*	Void

Table 2.8 One or more bottom profile samples can be added to the BottomProfile class.

2.1.4 Bottom type

The geo-acoustic properties of the bottom are coded by a single parameter in LybinTCPserver. Bottom types ranging from 1 to 9, where 1 represents a hard, rock type of bottom with low bottom reflection loss, while 9 represents a soft bottom with a high reflection loss. In addition, bottom types 0 and 10 have been added, representing *lossless* and *fully absorbing* bottoms, respectively.

Bottom type is one of three options for modelling the bottom loss. Bottom type is the default choice if both UseMeasuredBottomLoss and UseRayleighBottomLoss are set to false, which also are their default setting. Both these parameters can be found in the LybinModelData class.

Bottom type is the default of the four possible options to calculate bottom reverberation. LybinTCPserver will use the given bottom type when the TypeOfRevNoiseCalculation parameter in LybinModelData class is set to 0 (Calculate bottom reverberation from bottom types).

A range dependent bottom type sample can be added to the BottomType class as listed in *Table 2.10.* . Each range dependent bottom type is given as a StartStopSampleDouble as listed in *Table 2.9*

Class	Parameter	Type	Default value	Unit
StartStopSampleDouble	**Start**	Integer	0	m
	Stop	Integer	0	m
	Value	Double	2	

Table 2.9 The StartStopSampleDouble class contains Start, Stop and Value.

Function	Type
Add(StartStopSampleDouble bottomTypeSample) *Add a bottom type start stop sample.*	*Void*

Table 2.10 Add a start stop bottom type sample to the BottomType class.

2.1.5 Lamberts coefficient

Lamberts rule is one of four possible options to calculate bottom reverberation. According to Lamberts rule, the back scattering coefficient is given by:

$$\sigma(\theta) = \mu \sin^2 \theta \tag{2.1}$$

Where σ is the back scattering coefficient, θ is the incident grazing angle and μ is the Lamberts coefficient.

The input parameter LambertsCoefficient is range dependent, and needs appurtenant start and stop values. If LambertsCoefficient is to be used, the parameter TypeOfRevNoiseCalculation has to be set to 3, in order to use Lamberts rule in the calculation of the bottom reverberation. The parameter TypeOfRevNoiseCalculation can be found in the LybinModelData class.

The range dependent Lamberts coefficient can be added using the Add function, which is listed in *Table 2.12*. Each coefficient is given as a StartStopSampleDouble as listed in *Table 2.11*.

Class	Parameter	Type	Default value	Unit
StartStopSampleDouble	**Start**	Integer	0	m
	Stop	Integer	0	m
	Value	Double	0	dB

Table 2.11 The StartStopSampleDouble class contains Start, Stop and Value.

Function	Type
Add(StartStopSampleDouble lambertsCoeffSample) *Add a range dependent Lamberts coeffcsient sample.*	*Void*

Table 2.12 Add a range dependent Lamberts coeffcsient sample.

2.1.6 Ocean

The parameters in the ocean class represent the ocean environment and targets within the sea. All the parameters in the ocean class are listed in *Table 2.13*.

Both Ambient noise and target strength can either be given as a fixed parameter, –or it can be calculated from the given environmental input. Which one of these alternatives to be used is decided by the parameters NoiseCalculation and UseMeasuredTargeStrengt -in LybinModelData.

Parameter	Type	Default value	Unit
AmbientNoiseLevel *Noise from ambient sources.*	Double	50	dB
PH *pH level in the sea water.*	Double	8	
PrecipitationType *Type of precipitation in the area.* *0: No precipitation* *1: Light rain* *2: Heavy rain* *3: Hail* *4: Snow*	Enum	0	

Parameter	Type	Default value	Unit
ReverberationZone *The reverberation zone that the target is within, relative to the ship. This parameter is only applicable to CW-pulses.* *0: MainLobe* *1: Typical* *2: NoReverb*	Enum	1	
ShipDensity *Density of ship traffic in the area of the calculation. The ship density can vary from 1 (low) to 7 (high).*	Double	4	
SurfaceScatterFlag *True: Surface reflected ray angles will be modified in order to simulate rough sea scattering.* *False: Rays hitting the sea surface will be reflected specularly, as from a perfectly smooth surface.*	Boolean	true	
TargetAspectAngle *Aspect angle of target.*	Double	0	Deg
TargetCource *Cource of target.*	Double	0	Deg
TargetSpeed *Speed of target.*	Double	10	m/s
TargetStrength *Target echo strenght.*	Double	10	dB

Table 2.13 Parameters in the Ocean class.

2.1.7 Rayleigh bottom loss

In order to calculate the bottom loss more accurately, a Rayleigh bottom loss model is included. The Rayleigh bottom loss is based on the physical parameters: bottom attenuation, bottom sound speed and density ratio. In order to relate these bottom parameters to other bottom

models, the sound speed in the water at bottom depth is assumed to be 1500 m/s. This sound speed is only used in the calculation of bottom loss, and will not influence any other part of the model. The Rayleigh bottom loss is not range dependent. The parameters in the RayleighBottomLoss calss are listed in *Table 2.14*.

Class	Parameter	Type	Default value	Unit
RayleighBottomLoss	**BottomAttenuation**	Double	0,5	dB/wavelength
	BottomSoundSpeeed	Double	1700	m/s
	DensityRatio	Double	2	

Table 2.14 Parameters in the RayleighBottomLoss class.

In order to make LybinTCPserver calculate and use Rayleigh bottom loss, the UseRayleighBottomLoss parameter in LybinModelData class must be set to true. This parameter will overrule the parameter UseMeasuredBottomLoss if there is any conflict between the settings of the two.

2.1.8 Reverberation and noise measurements

The ReverberationAndNoiseMeasurements can consist of any number of measurements with corresponding ranges. To find values for the ranges not given as measurements, LybinTCPserver uses linear interpolation.

Reverberation and noise measurements are an optional choice where one uses measured values instead of letting LybinTCPserver estimate reverberation and noise. LybinTCPserver will only use the reverberation and noise measurements values given if the TypeOfRevNoiseCalculation parameter in LybinModelData class is set to 2 (Use measured reverberation and noise data).

The ReverberationAndNoiseSample can consist of any reverberation and noise samples containing range and depth values as listed in *Table 2.15*. The samples are added using the Add function as listed in *Table 2.16.*

Class	Parameter	Type	Default value	Unit
ReverberationAndNoiseSample	**Depth**	Double	0	m
	Scatter	Double	80	dB

Table 2.15 The ReverberationAndNoiseSample contains range and depth.

Function	Type
Add(ReverberationAndNoiseSample reverberationAndNoiseSample) *Add a reverberation and noise sample containing depth and scatter.*	Void

Table 2.16 One or more ReverberationAndNoiseSamples can be added .

2.1.9 Sound speed

The sound speed is a function of both range and depth. Since the sound speed is most often measured as depth dependant profiles, the SoundSpeed class can contain multiple sound speed profiles, representative of different ranges. The sound speed profiles contain sound speed samples holding the parameters temperature, salinity and sound speed for a given set of depths, as listed in *Table 2.17*.

One or more sound speed profiles can be added to the SoundSpeed class using the function Add, as listed in *Table 2.18*.

Class	Parameter	Type	Default value	Unit
SoundSpeedSample	**Depth**	Double	0	m
	SoundSpeed	Double	1480	m/s
	Temperature	Double	7,36	°C

Class	Parameter	Type	Default value	Unit
	Salinity	Double	35	parts per thousand
SoundSpeedProfile	**Start**	Integer	0	m
	Stop	Integer	0	m
	Latitude	Double	0	deg N
	Longitude	Double	0	deg E
	SoundSpeedSamples	List<SoundSpeedSample>		

Table 2.17 A sound speed profile contains one or more sound speed samples.

Function	Type
Add(SoundSpeedProfile soundSpeedProfile) *Add a sound speed profile.*	Void

Table 2.18 One or more sound speed profiles can be added to the SoundSpeed class.

2.1.10 Surface back scatter

Surface back scatter is the fraction of energy that is scattered back towards to the receiver when a ray hits the sea surface. The surface back scattering is generally a function of wind speed, wave height, grazing angle and frequency. A dataset representing surface back scattering coefficients is entered into LybinTCPserver, giving backscattering coefficients (in dB) for a set of grazing angles. Based on the values, LybinTCPserver interpolates to create backscattering coefficients for equidistantly spaced grazing angles. The back scattering coefficients are given as dB per square meter.

Surface back scatter is an optional choice to calculate surface reverberation. LybinTCPserver will only use the surface back scatter values given if the UseMeasuredSurfaceBackScatter parameter in LybinModelData class is set to true.

A start stop double list containing range dependent surface back scatter values can be added to the SurfaceBackScatter class as listed in *Table 2.19* and *Table 2.20*.

Class	Parameter	Type	Default value	Unit
DoubleSample	**Data**	Double	0	dB
	Key	Double	0	deg
StartStopDoubleList	**Start**	Integer	0	m
	Stop	Integer	0	m
	Samples	List<DoubleSample>		

Table 2.19 Surface back scatter start stop list contains start, stop and surface back scatter double samples.

Function	Type
Add(StartStopDoubleList surfaceBackScatterTable) *Add a range dependent surfaceback scatter table.*	*Void*

Table 2.20 Add a range dependent surface loss table to the SurfaceBackScatter class.

2.1.11 Surface loss

Surface loss is the fraction of energy that is lost after the sound has been reflected from the ocean surface, usually expressed in dB. The surface loss is also referred to as *forward scattering* in underwater acoustic terminology. Surface loss is generally a function of wind speed, wave height, gracing angle and frequency. A dataset representing surface loss is entered into LybinTCPserver, giving surface loss (in dB) for a set of grazing angles. Based on the values, LybinTCPserver interpolates to create loss values for equidistantly spaced grazing angles.

The parameter UseMeasuredSurfaceLoss tells LybinTCPserver to use SurfaceLossTable instead of calculating the surface loss. UseMeasuredSurfaceLoss must be set to true if one wants to use predefined surface loss values in LybinTCPserver. UseMeasuredSurfaceLoss can be found in the LybinModelData class.

A start stop double list containing range dependent surface loss values can be added to the SurfaceLoss class as listed in *Table 2.21* and *Table 2.22*.

Class	Parameter	Type	Default value	Unit
DoubleSample	**Data**	Double	0	dB
	Key	Double	0	deg
StartStopDoubleList	**Start**	Integer	0	m
	Stop	Integer	0	m
	Samples	List<DoubleSample>		

Table 2.21 Surfaceloss start stop list contains start, stop and surface loss double samples.

Function	Type
Add(StartStopDoubleList surfaceLossTable) *Add a range dependent surface loss table.*	*Void*

Table 2.22 Add a range dependent surface loss table to the SurfaceLoss class.

2.1.12 Surface reflection angle

Predefined surface reflection angles can be set using the Add function as listed in *Table 2.24*. Each surface reflection is given as a StartStopSampleDouble as listed in *Table 2.23.*

Surface reflection angle is an optional parameter that can be used to completely control the surface reflection of each ray in a simulation. If surface reflection angle is to be used the parameter UseSurfaceReflectionAngles must be set to true. The parameter UseSurfaceReflectionAngles can be found in the LybinModelData class.

Class	Parameter	Type	Default value	Unit
StartStopSampleDouble	**Start**	Integer	0	m
	Stop	Integer	0	m
	Value	Double	0	deg

Table 2.23 The StartStopSampleDouble class contains Start, Stop and Value.

Function	Type
Add(StartStopSampleDouble surfaceReflectionSample) *Add a range dependent surface reflection sample.*	*Void*

Table 2.24 Add a start stop surface reflection sample to the SurfaceReflection class.

2.1.13 Target strength

It is possible to include tables of target strength values. Each table consists of target strength values as a function of aspect angle. The aspect angle can be from 0-359°. If only values less than 180° are given in the table, the target strength values are reflected symmetrically through the longitudinal axis of the target. Each target strength table has a valid frequency range with a given minimum and maximum frequency.

The actual aspect angle to be used in the simulation is given in degrees by the parameter TargetAspectAngle. Whether LybinTCPserver shall find target strength from the table or use the parameter TargetStrength, is given by the parameter UseMeasuredTargeStrength. If UseMeasuredTargeStrength is true, the parameter TargetStrength will be updated with the target strength value that was actually used, found in the table based on frequency and target aspect angle.

A start stop double list containing frequency dependent target strength values can be added to the TargetStrengtrh class as listed in *Table 2.25* and *Table 2.26*.

Class	Parameter	Type	Default value	Unit
DoubleSample	**Data**	Double	0	dB
	Key	Double	0	deg
StartStopDoubleList	**Start**	Integer	0	m
	Stop	Integer	0	m
	Samples	List<DoubleSample>		

Table 2.25 Target strength start stop list contains start, stop and target strength double samples.

Function	Type
Add(StartStopDoubleList targetStrengthTable) *Add a range dependent target strength table.*	*Void*

Table 2.26 Add a frequency dependent target strength table to the TargetStrength class.

2.1.14 Volume back scatter

Volume back scatter is the fraction of energy scattered back towards the receiver from the sea volume. Scattering elements in the sea volume can be particles or organic life, like plankton, fish or sea mammals. The volume back scatterers are not distributed uniformly in the sea, and may vary considerably as a function of depth, range and time of the day. In LybiTCPserver, the volume back scatter is given as a profile of back scattering coefficients as a function of depth. Scatter values for the depths between data points are calculated using linear interpolation. The influence region of each profile is determined from the corresponding start range and stop range values.

The volume back scatter profiles contain volume back scatter samples, as listed in *Table 2.27*.

One or more volume back scatter profiles can be added using the function Add, as listed in *Table 2.28*.

Class	Parameter	Type	Default value	Unit
Volume BackScatter Sample	**Depth**	Double	0	m
	Scatter	Double	80	dB
Volume BackScatter Profile	**Start**	Integer	0	m
	Stop	Integer	0	m
	Latitude	Double	0	deg N
	Longitude	Double	0	deg E
	VolumeBackScatter Samples	List<VolumeBackScatter Sample >		

Table 2.27 A VolumeBackScatterProfile contains one or more VolumeBackScatterSamples.

Function	Type
Add(VolumeBackScatterProfile volumeBackScatterProfile) *Add a volume back scatter profile.*	Void

Table 2.28 Add a volume back scatter profile.

2.1.15 Wave height

The WaveHeight consist of wave height samples containing range and height values as listed in *Table 2.29*. The samples are added to the WaveHeight class using the Add function as listed in *Table 2.30*.

Wave height is an optional parameter to wind speed. If wave height is to be used the parameter UseWaveHeight found in the LybinModelData class must be set to true.

Class	Parameter	Type	Default value	Unit
StartStopSampleDouble	**Start**	Integer	0	m
	Stop	Integer	0	m
	Value	Double	0	m

Table 2.29 The StartStopSampleDouble class contains Start, Stop and Value.

Function	Type
Add(StartStopSampleDouble waveHeightSample) *Add a wave height sample.*	*Void*

Table 2.30 Add a start stop wave height sample to the WaveHeight class.

2.1.16 Wind speed

The wind speed consists of wind speed samples containing range and speed values as listed in *Table 2.31*. The samples are added to the WindSpeed class using the Add function as listed in *Table 2.32*.

Class	Parameter	Type	Default value	Unit
StartStopSampleDouble	**Start**	Integer	0	m
	Stop	Integer	0	m
	Value	Double	0	m/s

Table 2.31 The StartStopSampleDouble class contains Start, Stop and Value.

Function	Type
Add(StartStopSampleDouble windSpeedSample) *Add a wind speed sample.*	*Void*

Table 2.32 Add a start stop wind speed sample to the WindSpeed class.

2.2 Platform

The platform class contains all the relevant information about the platform holding the sonar. The platform is most often a ship, but can also be a helicopter or a buoy. The parameters in the platform class are listed in *Table 2.33*.

Parameter	Type	Default value	Unit
Latitude *Actual latitude of platform.*	Double	0	deg
ShipCourse *Platform course relative to north.*	Double	0	deg
SelfNoise *Noise from the platform that holds the sonar.*	Double	50	dB
SelfNoisePassive *Noise from the platform that holds the sonar. To be used in calculations for passive sonars.*	Double	50	dB
Sensor *All the sensor data to be used in the calculation.*	Sensor		
Speed *Speed of the platform that holds the sonar.*	Double	10	Knots

Table 2.33 Parameters in the platform class.

2.2.1 Sensor

The sensor class contains all the relevant information about the sonar. The parameters in the sensor class are listed in *Table 2.34*.

Parameter	Type	Default value	Unit
BeamWidthHorizontal *Horisontal beam width of the sonar.* *If BeamWidthHorizontal is to be used, the parameter UseMeasuredHorizontalBeamWidth must be set to true.*	Double	20	Degrees
BeamWidthReceiver *Vertical beam width of the receiving part of the sonar.*	Double	15	Degrees
BeamWidthTransmitter *Vertical beam width of the transmitting part of the sonar.*	Double	15	Degrees
CalibrationFactor *The parameter is on the interface, but are not yet implemented or used in the calculations.*	Double	0	dB
Depth *Depth of the sonar.*	Double	5	Meters
DetectionThreshold *The strength of the signal relative to the masking level necessary to see an object with the sonar.*	Double	10	dB
DirectivityIndex *The sonars ability to suppress isotropic noise relative to the response in the steering direction.*	Double	20	dB
Frequency *Centre frequency of the sonar.*	Double	7000	Hz
IntegrationTimePassive *Integration time for the passive sonar.*	Double	1	Seconds

Parameter	Type	Default value	Unit
PassiveBandWidth *Band width of the passive sonar.*	Double	100	Seconds
PassiveFrequency *Centre frequency of the passive sonar.*	Double	800	Hz
PassiveProcessinGain *Gain of the passive sonar.*	Double	0	dB
Pulse *All the pulse data to be used in the calculation.*	Pulse		
SideLobeReceiver *The suppression of the highest side lobe relative to the centre of the beam for the receiving sonar.*	Double	13	dB
SideLobeTransmitter *The suppression of the highest side lobe relative to the centre of the beam for the transmitting sonar.*	Double	13	dB
SonarTypePassive *Tells whether the passive sonar is broad- or narrowband.* *0: Narrowband* *1: Broadband*	Enumerator	0	
SourceLevel *Source level of the sonar.*	Double	221	dB
SourceLevelPassive *Source level of the possible target in the calculation for passive sonar.*	Double	100	dB
SystemLoss *System loss due to special loss mechanisms in the sea or sonar system, not otherwise accounted for.*	Double	0	dB
TiltReceiver *Tilt of the receiving part of the sonar.*	Double	4	Degrees

Parameter	Type	Default value	Unit
TiltTransmitter *Tilt of the transmitting part of the sonar.*	Double	4	Degrees

Table 2.34 Parameters in the sensor class.

2.2.1.1 Pulse

All the information about the pulse is gathered in the pulse class. All the access parameters in the pulse class are listed in *Table 2.35* below. The pulse class does not have any access functions.

Parameter	Type	Default value	Unit
EnvelopeFunc *Envelope function of the signal. Currently, only "Hann" is available.*	String	Hann	
FilterBandWidth *Filter bandwidth of the pulse.*	Double	100	Hz
FMBandWidth *Frequency modulation bandwidth of the pulse. Applicable for FM signals only.*	Double	100	Hz
Form *Pulse type:* *FM: Frequency modulated* *CW: Continuous wave*	String	FM	
Length *Pulse length.*	Double	60	Milliseconds

Table 2.35 Parameters in the pulse class.

3 Initiate calculation

The CalculateLybinModel function initiates a new instance of LybinTCPserver using the modelIndex returned from the function CreateLybinModel that is used to set the model data in the simulation. Both these functions are described in *Table 3.1*.

Function	Type
CalculateLybinModel(int modelIndex) *Start the calculation.*	LybinResults
CreateLybinModel(LybinModelDAta lybinModelData) *Send the model data to LybinTCPserver.*	Integer

Table 3.1 Functions for initiation of calculation.

4 Calculation results

4.1 Functions returning calculation results

The calculation results can be accessed through functions found in *Table 4.1*.

Function	Type	Unit
getCalculatedAmbientNoise(int modelIndex) *The ambient noise used in the calculations.*	Double	dB
getBottomReverberation(int modelIndex) *Calculated bottom reverberation values.*	List<double>	dB
getEchoLevel(int modelIndex) *Not yet implemented inside LybinCom. This object will not have any data.*	List<List<double>>	dB

Function	Type	Unit
getImpulseResponse(int modelIndex) *Get calculated impulse response.*	List<List<ImpulseResponsePoint>>	
getInterpolatedBottomProfile(int modelIndex) *Get the interpolated bottom profile.*	List<List<double>>	m
GetInterpolatedSoundSpeed(int modelIndex) *Get the smoothed and interpolated sound speed matrix.*	List<List<double>>	m
getMaskingLevel(int modelIndex) *Calculated masking level (total reverberation + noise after processing).*	List<double>	dB
getNoiseAfterProcessing(int modelIndex) *Calculated noise after processing.*	Double	dB
getProbabilityOfDetection(int modelIndex) *Calculated probability of detection.*	List<List<double>>	%
getRayTrace(int modelIndex) *Not implemented inside LybinCom. This object will not have any data.*	List<List<double>>	
getLybinModel(int modelIndex) *The model data used during the calculation.*	LybinModelData	
getSignalExcess(int modelIndex) *Calculated signal excess.*	List<List<double>>	dB
getSurfaceReverberation(int modelIndex) *Calculated surface reverberation.*	List<double>	dB
getTotalReverberation(int modelIndex) *Calculated total reverberation.*	List<double>	dB

Function	Type	Unit
getTransmissionLossReceiver(int modelIndex) *Calculated transmission loss from the target to the receiver.*	List<List<double>>	dB
getTransmissionLossTransmitter(int modelIndex) *Calculated transmission loss from the transmitter to the target.*	List<List<double>>	dB
getTravelTime(int modelIndex) *Returns the travel time paths calculated.*	List<List<TravelTimePoint>>	
getVisualRayTracet(int modelIndex) *Returns the visual ray trace paths calculated.*	List<List<VisualRayTracePoint>>	
getVolumeReverberation(int modelIndex) *Calculated volume reverberation.*	List<double>	dB

Table 4.1 Functions returning calculation results.

4.2 Impulseresponse point

All the parameters in the ImpulseResponsePoint class are listed in *Table 4.2*.

Parameter	Type	Unit
LongestTravelTime *Longest travel time.*	Double	s
MaxInitialAngle *Maximum initial ray angle.*	Double	deg
MeanInitialAngle *Mean initial ray angle.*	Double	deg

MinInitialAngle *Minimum initial ray angle.*	Double	deg
My *Mean arrival time – first arrival time.*	Double	s
Phase *Phase identifier.*	Integer	
RayFamilyCode *Ray family identifier. The ray family identifier represents the ray family's travel history, using the letter codes:* *s Surface reflection* *b Bottom reflection* *u Upper turning point* *l Lower turning point*	String	
S *Intensity loss.*	Double	dB
ShortestTravelTime *Shortest travel time.*	Double	s
Sigma *Arrival time standard deviation.*	Double	s
StandardDeviationInitialAngle *Standard deviation of initial ray angle.*	Double	deg

Table 4.2 Parameters in the ImpulseResponsePoint class.

4.3 Traveltime point

All the parameters in the TravelTimePoint class are listed in *Table 4.3*.

Parameter	Type	Unit
InitialAngle *Initial ray angle.*	Double	deg
Range *Range of travel time point.*	Double	m
Depth *Depth of travel time point.*	Double	m
TravelTime *Travel time from start to point.*	Double	s

Table 4.3 Parameters in the TravelTimePoint class.

4.4 Visual raytrace point

All the parameters in the VisualRayTracePoint class are listed in *Table 4.4*.

Parameter	Type	Unit
InitialAngle *Initial ray angle.*	Double	deg
Range *Range of travel time point.*	Double	m
Depth *Depth of travel time point.*	Double	m
TravelTime *Travel time from start to point.*	Double	s

Table 4.4 Parameters in the VisualRayTracePoint class.

References

1. E. Dombestein, and T. Jenserud, “Improving Underwater Surveillance: LYBIN Sonar performance prediction”, Proceedings of MAST 2010 – Rome, 2010.

2. K.T. Hjelmervik, S. Mjølsnes, E. Dombestein, T. Såstad and J. Wegge, “The acoustic raytrace model Lybin – Descriptions and applications”, UDT 2008, Glasgow, United Kingdom, 2008

3. E. Dombestein, "LYBIN 6.2 2200 - user manual", FFI Rapport 17/00412, 2017.

4. E. Dombestein, S. Mjølsnes, and F. Hermansen, "Visualization of sonar performance within environmental information," in Oceans 2013, Bergen, 2013.

5. E. Dombestein and F. Hermansen, "Integration of Sonar Performance Modelling in Sonar Operator Training, Mission Planning and High Risk Decisions," presented at the MSG-126, Washington DC, USA, 2014.

6. E. Dombestein, "LybinCom 6.2 - description of the binary interface," FFI-Rapport 2014/00511, 2014.

7. https://docs.microsoft.com/en-us/windows/win32/com/component-object-model--com--portal

8. https://thrift.apache.org/

About FFI

The Norwegian Defence Research Establishment (FFI) was founded 11th of April 1946. It is organised as an administrative agency subordinate to the Ministry of Defence.

FFI's MISSION

FFI is the prime institution responsible for defence related research in Norway. Its principal mission is to carry out research and development to meet the requirements of the Armed Forces. FFI has the role of chief adviser to the political and military leadership. In particular, the institute shall focus on aspects of the development in science and technology that can influence our security policy or defence planning.

FFI's VISION

FFI turns knowledge and ideas into an efficient defence.

FFI's CHARACTERISTICS

Creative, daring, broad-minded and responsible.

Om FFI

Forsvarets forskningsinstitutt ble etablert 11. april 1946. Instituttet er organisert som et forvaltningsorgan med særskilte fullmakter underlagt Forsvarsdepartementet.

FFIs FORMÅL

Forsvarets forskningsinstitutt er Forsvarets sentrale forskningsinstitusjon og har som formål å drive forskning og utvikling for Forsvarets behov. Videre er FFI rådgiver overfor Forsvarets strategiske ledelse. Spesielt skal instituttet følge opp trekk ved vitenskapelig og militærteknisk utvikling som kan påvirke forutsetningene for sikkerhetspolitikken eller forsvarsplanleggingen.

FFIs VISJON

FFI gjør kunnskap og ideer til et effektivt forsvar.

FFIs VERDIER

Skapende, drivende, vidsynt og ansvarlig.

FFIs organisasjon