WIPL-D Pro: What is New in v17?

New features/improvements introduced in v17 are:

1. **Full exploitation of Matrix Symmetry:**
   a. **Reduced matrix fill time**
   b. **Reduced matrix solution time**
2. **Improved GPU out-of-core solution:**
   a. **Matrix fill in parallel on multiple hard disks**
3. **Accelerated GPU matrix fill: up to 40%**
4. **New functionalities in massive RCS calculation ("Run As Avoided with Interpolation")**
   a. **Support for 3D angular interpolation**
   b. **Refined interpolation by Spline function**
   c. **New interpolation function: Rational function**
   d. **Error estimation**
5. **Refined accuracy of Periodic Boundary Condition**
6. **Sweep parallel run**
   a. **Concurrent simulation of multiple sub projects**
7. **Fast calculation of excitation matrix for Field Generators' arrays**
8. **STL editor**
   a. **Import of STL files**
   b. **Variety of edit options**
   c. **Decimation**
   d. **Quadrilateral meshing**
   e. **Export to STL files**
9. **Accelerated Topological Analysis**
10. **Easier handling of (very) complex structures**
    a. **Hide Selected Plates**
    b. **Select Layer**
    c. **Extended limits for "Manipulations"**
    d. **Fast Open/Save/Save As**

All the examples are simulated using two computers:

1. **Laptop:**
   a. CPU: Intel® Core(TM) i7-4720HQ @ 2.60 GHz (4 cores / 8 threads)
   b. RAM: 8 GB
2. **Workstation:**
   a. CPU: Intel® Xeon® CPU E5-2650 v4 @ 2.20 GHz (2 processors: 24 cores / 48 threads)
   b. RAM: 256 GB
   c. GPU: 4 x NVIDIA GeForce GTX 1080 Ti
   d. Disk: 7 x HDDs, Read/Write speed: 180 MB/s

1. Full exploitation of Matrix Symmetry (reduce the simulation time up to 50%)

By setting "Reduced" for Matrix inversion the MoM matrix equation is solved assuming that the MoM matrix is symmetrical with respect to the main diagonal. Thus the number of operations for matrix solve is halved when compared with the default case when symmetry of the matrix is not taken into account (i.e., Matrix inversion is set to "Normal").

Full exploitation of the matrix symmetry should enable halving of both, matrix fill time and matrix solve time. In the previous version (v16) the MoM matrix is symmetrical only for metallic structures made of plates, and the matrix symmetry is not exploited for reducing the matrix fill time.

In the new version (v17) two techniques are applied to symmetrize the MoM matrix in general case: a) matrix equilibration for structures containing dielectrics, and b) averaging of matrix elements symmetrically placed with respect to the main diagonal for structures containing wires.

In this way: a) the matrix fill time is reduced for arbitrary structure without wires, b) the matrix solve time is reduced for CPU matrix solve of smaller problems and GPU matrix solve of larger problems.

Example: Let us consider hyperboloidal lens antenna with two variants of excitation: a) realistic cylindrical waveguide antenna, and b) equivalent FG (Field Generator), as shown in Fig. 1.

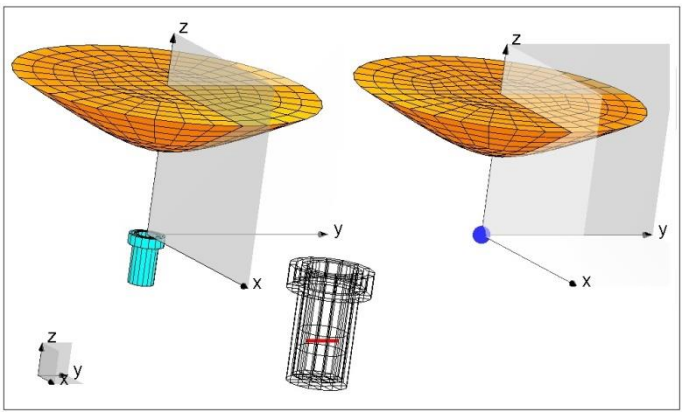

Fig. 1. Model of lens antenna exicted by a) cylinrical waveguide feed, and b) equivalent field generator.

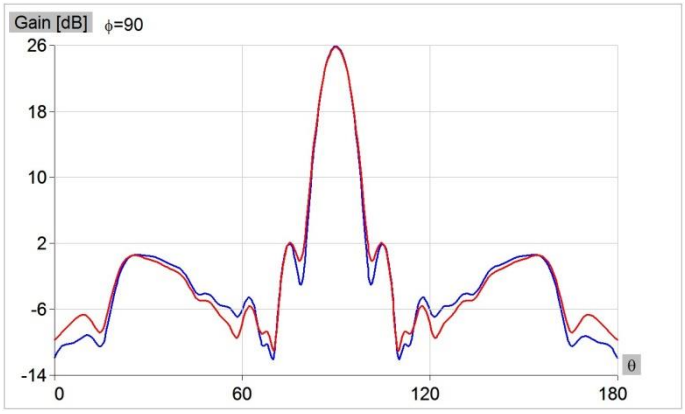

Fig.2. Radiation pattern in plane ϕ = 90° obtained by realistic feed (red) and equivalent model (blue).

Fig. 2 shows radiation pattern in plane ϕ = 90° obtained by both feed models. Good agreement between the results is observed. The realistic feed contains wires so that matrix symmetry can be used only to reduce the matrix solve time. The equivalent feed does not contain wires, so that matrix symmetry can be used to reduce not only the matrix solve time, but also the matrix fill time.

Let us consider three models with equivalent feed, A, B, and C, with different number of unknowns. Table 1 compares matrix fill times and matrix solve times for all three models when the matrix symmetry is exploited with that when the matrix symmetry is not exploited. The simulation is performed on the laptop. It is seen that matrix fill time is reduced for 35%, 38%, and 40%, and matrix solve time, for 31%, 36%, and 47%, respectively.

Table 1. Matrix fill/solve times for models A-C, when matrix symmetry is (is not) exploited.

Model (Number of Unknowns)	*Matrix Symmetry Exploited*	*Matrix fill time [s]*	*Matrix solve time [s]*
A (11,488)	No	18,45	18,86
A (11,488)	Yes	11.97	13.04
B (16,720)	No	37.42	60,19
B (16,720)	Yes	23.13	38,49
C (23, 008)	No	70,01	181.14
C (23,008)	Yes	42.17	95,05

2. Improved GPU Out-Of-Core Solution
(Matrix fill in parallel on multiple hard disks)

Starting from the very first version of WIPL-D GPU Solver hard disks I/O operations during the matrix inversion procedure were accelerated by parallel usage of multiple hard disks. Nevertheless, in all previous versions, during the matrix fill process, system matrix was saved on only one hard disk, the one where the model is saved. In the case when the model is electrically very large, and system matrix cannot be stored on only one disk, RAID 0 configuration of multiple disks was recommended.

In the new version of WIPL-D GPU Solver, system matrix is saved on multiple hard disks in matrix fill procedure. The matrix is equally distributed across all hard disks selected for simulation. This way:

- There is no requirement for a hard disk large enough to enable saving of entire system matrix, but the total size of multiple hard disks should be large enough.
- Input and output operations to and from hard disks are significantly accelerated, as there is no slow down due to limited bandwidth of RAID controller. The acceleration can be observed in matrix storage time in matrix fill-in procedure, and especially in matrix inversion time, as communication with hard disks is very intensive during this stage of simulation process.

Example: The achieved acceleration is illustrated on the model of F16 fighter, as shown in Fig. 3. The monostatic RCS is calculated at frequency of 3 GHz, in the plane ϕ = 0°, in 721 directions. Number of unknowns: 320,822.

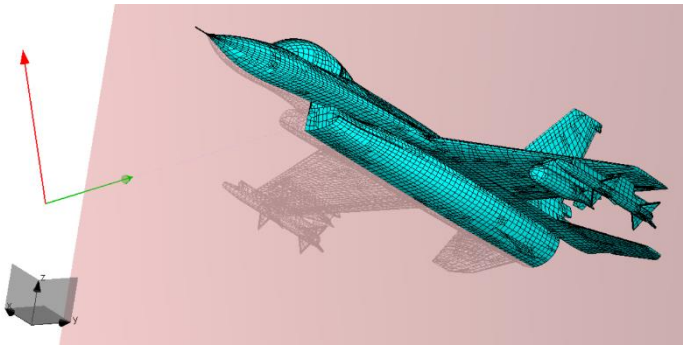

Fig. 3. Model of F-16 fighter illuminated by plane wave.

This simulation was performed on the workstation. Comparison of matrix storage, inversion and total simulation time for the previous version, where disks were connected in RAID-0 configuration, and the new version with separate drives is given in the Table 2. It is seen that matrix storage time, matrix solution time and total simulation time are reduced for 27%, 20%, and 18%, respectively.

Table 2. Times for certain simulation phases and total simulation time in v16 and v17.

Version	*Matrix Storage time [s]*	*Matrix Solution time [s]*	*Total simulation time [s]*
v16	590	7,011	9,862
v17	433	5,585	8,100

3. Accelerated GPU Matrix Fill

Matrix fill is the most time consuming phase in 3D EM simulation of small and medium sized problems. From the very first version of WIPL-D Pro (3D EM solver) there are continuous efforts to accelerate this phase, including parallelization on both CPU and GPU.

In the new version (v17) matrix fill performed on GPUs is accelerated up to 40% when compared with that in previous version (v16). In this way, matrix fill based on GPUs becomes choice of preference for expansion orders equal to n = 3 and lower (i.e. patch sizes lower than wavelength).

Example: 2D array of m x m cube scatterers is set, where m = 10,12,15,20,30,60, as shown for m = 10 in Fig. 4. By decreasing the cube side, the expansion orders are respectively set to n = 1,2,3,4,5,6, so that total number of unknowns in all cases is N = 43,200. The simulations are performed at workstation for various numbers of CPU cores and GPUs in both, v16 and v17. Table 3 and 4 gives the matrix fill time for various GPUs in v16 and 17, respectively. Since matrix fill on CPU is practically the same in both versions, the corresponding times are shown for both versions in Table 5.

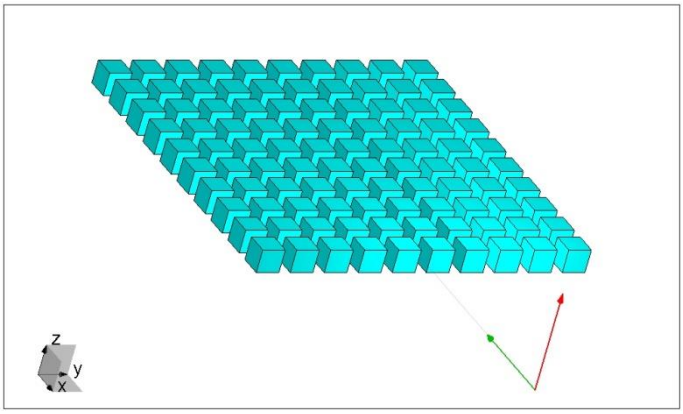

Fig. 4. Array of cube scatterers.

Table 3. Matrix fill time in seconds for various expansion orders and various numbers of GPUs obtained in v16.

Expansion Order	1 GPU	2 GPUs	3 GPUs	4 GPUs
1	142.29	142.22	135.57	131.57
2	37.57	32.16	30.23	29.75
3	31.06	24.99	22.31	21.79
4	33.92	25.12	21.79	20.66
5	37.97	27.14	23.30	22.01
6	44.38	31.11	25.83	24.01

Table 4. Matrix fill time in seconds for various expansion orders and various numbers of GPUs obtained in v17.

Expansion Order	1 GPU	2 GPUs	3 GPUs	4 GPUs
1	93.40	85.98	85.24	81.02
2	31.10	22.18	21.26	19.86
3	27.75	19.21	17.78	15.51
4	30.86	20.53	18.31	15.84
5	37.49	23.74	21.67	18.28
6	42.59	27.28	21.96	20.15

Table 5. Matrix fill time in seconds for various expansion orders and various numbers of CPUs obtained in v16/v17.

Expansion Order	4 cores	8 cores	12 cores	16 cores	20 cores
1	915.98	606.57	497.46	384.40	426.96
2	143.66	77.41	51.60	49.14	46.80
3	80.43	42.88	30.48	27.63	27.09
4	64.45	34.55	24.58	23.55	21.39
5	60.44	32.42	22.90	21.19	21.06
6	58.04	31.22	22.39	21.81	20.53

The tables confirm that “matrix fill” on GPUs by v17 is accelerated up to 40% when compared with that by v16, and that GPUs are preferable over CPUs for n < 4. Note that, whatever is number of CPU cores or GPUs, the GPU matrix fill for order n = 1 is more than four times faster than that of CPU.

4. New functionalities in RCS calculation (“Run As Avoided with Interpolation”)

In massive RCS calculation it is often required that number of excitations at one frequency is even far away higher than the number of problem unknowns. In these cases simulation time can be even few orders of magnitude higher than time needed for one excitation (i.e. time for LU decomposition). This type of simulation can be dramatically accelerated if RCS calculation is performed in properly limited number of directions followed by “Run As Avoided with Interpolation”. In addition to interpolation along angle of incoming wave direction, it is possible to perform interpolation in the frequency range.

a) Support for 3D angular interpolation

In the previous version (v16) angular interpolation is enabled only along one spherical angle. In the new version (v17) the applicability of this option is extended to simultaneous interpolation along both spherical angles.

Example: Let us consider dielectric cube scatterer of side a = 1 m, relative permittivity of $\varepsilon_r = 4$, at frequency of f = 355 MHz. Monostatic RCS is calculated in a full range of directions in 3D space ($-180° < \phi < 180°$, $-90° < \theta < 90°$), in a limited number of directions, $n_\phi \times n_\theta = 49 \times 25 = 1225$. (Step along each angle is 7.5 degrees.) Interpolation is performed using $n_\phi \times n_\theta = 361 \times 181 = 65341$ directions, i.e., with the step of 1 degree. Figs. 5 and 6 show the RCS before and after interpolation. Total simulation times in the cases with/without interpolation are given in Table 6.

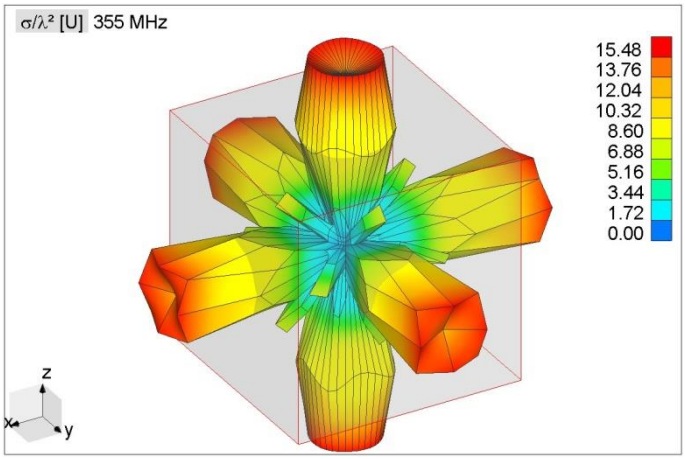

Fig. 5. 3D graph of monostatic RCS (before interpolation)

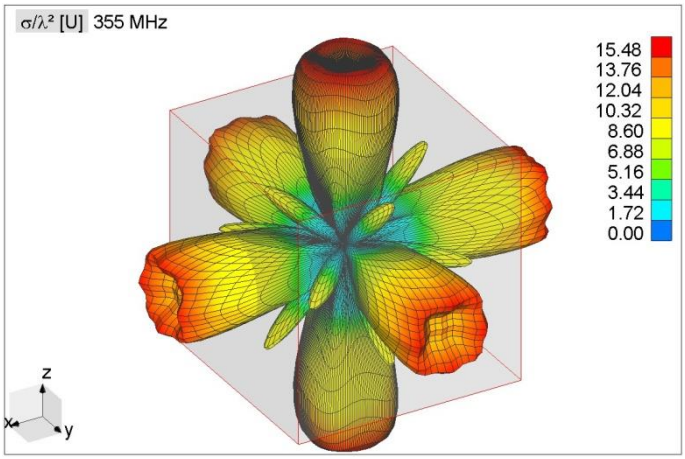

Fig. 6. 3D graph of monostatic RCS (after interpolation)

Table 6. Times for certain simulation phases and total simulation time in v16 and v17.

Total Solution Time WITH interpolation [s]	*Total Solution Time WITHOUT interpolation [s]*
4.99 + 80.34 = 85.33	231.10

b) Refined interpolation by Spline function

Two overlapping spline functions are combined using optimal weights, thus providing smooth curve for results.

c) New interpolation function: Rational function

Interpolation in frequency domain using spline functions is shown to be less successful than that in domain of angle directions. It is shown that in some cases interpolation by rational functions is more effective than that by splines. Hence, the possibility of using rational functions instead of splines is added in the new version (v17).

In particular, it is shown that usage of rational functions is more effective if the original problem with limited number of directions and frequencies is solved using orthogonal or ultra-high bases functions instead of classic or modified ones.

d) Error estimation

Two overlapped interpolation functions are used to estimate the interpolation error. Usually, such calculated error is smaller for rational functions than for splines. In spite of that the spline is preferable choice for interpolation along angle directions, while in the case of interpolation in the frequency range, it varies from example to example.

5. Refined Accuracy in Periodic Boundary Condition

It is found in the previous version (v16) that in some cases the convergence of the results is relatively slow with increase of the integral accuracy. The reason for such behavior is that the field due to cells neighboring to the main cell can also exhibit the quasi-singular behavior.

In the new version (v17) this problem is resolved by detecting all (quasi) singular points and performing all necessary extractions of singularities, as well as their analytical evaluations.

6. Sweep parallel run
(Concurrent simulation of multiple sub-projects)

WIPL-D numerical kernel is CPU and GPU accelerated, with support for an arbitrary number of CPU threads and GPUs. Efficiency of multi-GPU acceleration is very good for electrically large problems, especially for the problems with more than 100,000 unknowns. On the other hand, efficiency of multi GPU acceleration for electrically small problems and problems of moderate size is often rather low. Efficiency of CPU acceleration also depends on size of the problem, and increases with increasing the number of unknown coefficients. For small problems and large number of utilized CPU threads, high efficiency of parallelization cannot be expected.

Starting from WIPL-D Pro v13 the option named "Frequency Parallel Run" is available in WIPL-D software package. This option enables concurrent simulation of multiple frequency points. The similar approach is now introduced for Sweep option. The basic idea that stands behind is to enable parallel simulation of different projects for small or mid-size models, where each simulation would use its own part of hardware resources (threads, GPUs, etc.). It leads to full utilization of available hardware resources with very high parallelization efficiency.

After one chooses the option "Sweep Parallel Run" all projects that should be simulated in a Sweep run are automatically created. Afterwards groups of the projects are simulated concurrently, where each project in a group utilize its own set of CPU threads and GPUs. Finally, after the simulation of all sweep subprojects is finished, output files are automatically merged.

Example: Let us start from the example in Section 1: lens antenna excited by the equivalent FG. By the book, FG is placed in the focal point, at distance Zfeed = 22.5 mm from the hyperboloid. The antenna gain can be further increased by moving it a bit along the antenna main axis. The optimal position can be found using Sweep option.

The first experiment is performed on the laptop using 8 steps, from Zfeed = 20.5 mm to Zfeed = 27.5 mm. Results for gain are shown in Fig. 7. The sweep is performed using 1, 2, and 4 runs in parallel. Corresponding simulation times are given in Table 7.

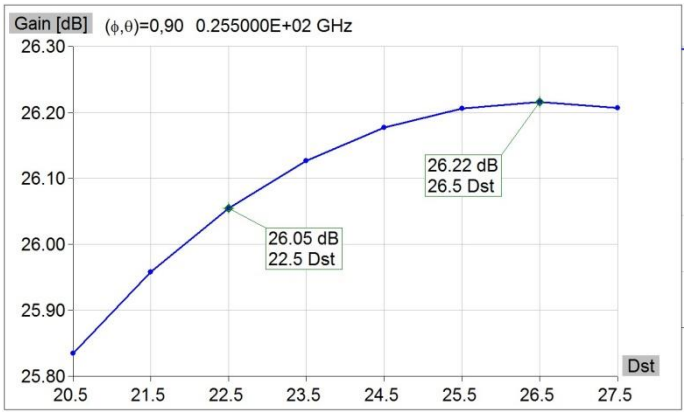

Fig. 7. Gain versus distance of FG (Dst) from lens.

Table 7. Total simulation times for sweep performed at the laptop using various number of runs in parallel.

Number of Parallel runs	*1*	*2*	*4*
Total simulation time [s]	213	201	185

The second experiment is performed on the workstation using 24 steps. The sweep is performed using 1, 2, 4, 6, 8, 12, and 24 runs in parallel. Corresponding simulation times are given in Table 8.

Table 8. Total simulation times for sweep performed at the workstation using various number of runs in parallel.

Number of Parallel runs	*1*	*2*	*4*	*6*	*8*	*12*	*24*
Total simulation time [s]	237	152	123	115	112	109	108

It is seen that at the laptop reduction in simulation time is relatively small, while at the workstation it is significant. It is explained by the fact that efficiency of single run is much higher at the laptop than at the workstation.

7. Fast Calculation of Excitation Matrix for Field Generators Arrays

Field generators can be successfully utilized as an approximation of realistic antenna arrays and very efficient characterization of array and radome performances for different scan angles. Nevertheless, realistic antenna arrays consists of a high number of radiating elements, while real life radomes are electrically very large and consists of multiple layers. On the other hand, precise characterization of a radome structure requires simulation of multiple scan angles of an array which is placed below the radome. All above-mention facts lead to the conclusion that calculation of excitation matrix for such scenarios is very challenging. That is exactly what we see in WIPL-D simulations, where calculation of excitation matrix for mentioned scenarios represents the main part of total simulation time.

By applying special acceleration and parallelization techniques this time is significantly reduced in the new version of WIPL-D Pro.

It will be illustrated on the radome structure shown in the Fig. 8. The radome has ellipsoidal shape, where lengths of semi-axes are $a_x = a_y = 1.25$ m and $a_z = 2.85$ m, radome thickness is equal to 5 mm, while relative permittivity of radome material is equal to 2. Frequency of interest is 10 GHz. Two (A)Symmetry planes are utilized in order to reduce number of unknown coefficients. Number of unknowns is about 300,000. FG array consists of 900 elements, and 25 steering angles is calculated.

The simulation was performed on the workstation.

Excitation matrix fill-in time is reduced from 13.25 hours to 2.8 hours, per (A)Symmetry project. Therefore, we achieved acceleration of almost 5 times.

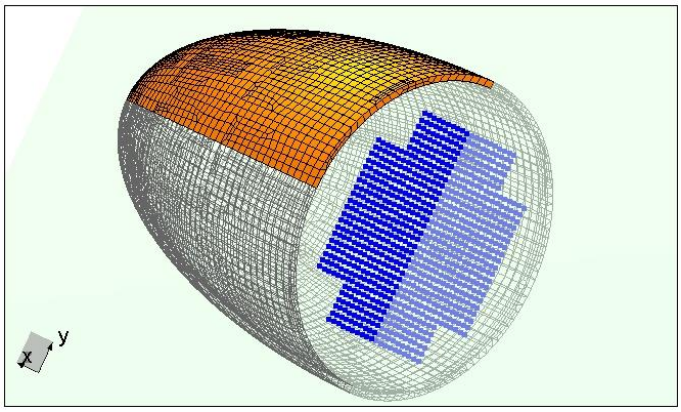

Fig. 8. Electrically large radome above a FG array.

8. STL editor

An STL file contains information about 3D objects. It describes the triangulated surfaces of such objects, where each triangle is determined by Cartesian coordinates of its three vertices.

With the development of 3D printing and computer-aided manufacturing, .stl files have gained in importance. STL files have become extremely widespread, and numerous computer-aided design (CAD) systems enable export of projects in STL file format. In order to enable import of .stl files, as well as manipulation over STL models and its conversion to WIPL-D native format (IWP), a new module named "STL Editor" is included in the WIPL-D Pro environment.

e) Import of .stl files

Opening and combining of multiple STL files (both binary and ASCII encoding) is supported in the new version. Opening .stl file is a process that results in creation of an STL project in WIPL-D Pro environment. The example of the STL project opened in WIPL-D Pro environment is presented in Fig. 9.

Models recorded in the .stl format usually consist of a huge number of triangles and in many cases have a number of topology and geometry irregularities. For those reasons, before performing the necessary conversion of a triangle-based model into a quad-based WIPL-D mesh, some modifications need to be made over the original STL model.

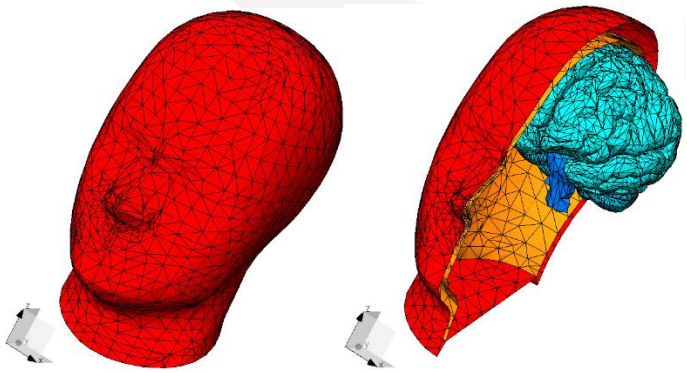

Fig. 9. Opened STL model representing the head and the brain (consisted of 15535 triangles, mostly for brain), full model (left) and model with hidden outer parts (right).

f) Variety of Edit options

As it may happen that an STL model has some irregularities, like unnecessary openings or shifted triangles or vertices, STL editor offers healing and reconstructing tools, which allow modifications of existing triangles as well as the creation of new ones.

- STL crop is the option that allows users to cut the model using arbitrary plane and delete the parts of the model that are placed below or above the plane. An example of crop operation is shown in Fig. 10.

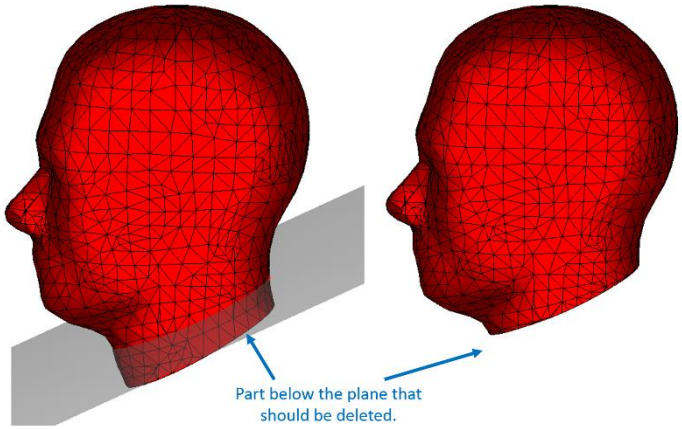

Fig. 10. STL crop operation used to remove part of the structure.

- STL fill is the option created for closing open parts of the structure, where all the nodes located on the laminar loop are coplanar.

Finally, it is possible to define domain characteristics for each individual triangle. As the number of triangles in a project can be huge, in order to make it easier to set domain characteristics, two options are introduced:

- Select Layer – invokes selection of connected triangles. Selection is limited by the locations

where more than two triangles share a common edge.

- Hide Selected Plates – as the name states, hides the marked plates.

Combining these two options facilitates the setting of domain specifications to triangles, especially when working with multilayer multi-domain structures.

g) Decimation

WIPL-D decimation tool enables simplification of the models, by applying two different decimation techniques: Deviation controlled, and Angle controlled decimation.

- Deviation controlled decimation is based on the control of the maximum distance of a triangle included in the new (simplified) structure from the corresponding nodes in the starting (original) model.
- Angle controlled decimation is based on the control of the angle between normals of neighboring triangles. It also allows users to limit the maximum size of a single element in the simplified structure to a specified value.

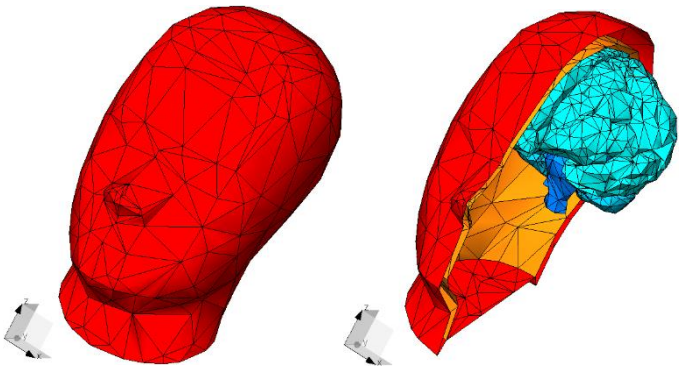

Fig. 11. Decimated model of a human head with brain inside (consisted of 4473 triangles), full model (left) and model with hidden outer parts (right).

The model created by applying the decimation tool on the model shown in Fig. 9 can be seen in Fig. 11. Original model consisted of 15535 triangles, while the newly created one includes 4473 triangles.

h) Quadrilateral meshing

Quadrilateral-based .iwp file format is the WIPL-D native format. In order to run EM simulation in WIPL-D solver, STL project has to be converted to IWP file. For such conversion we offer three meshing mechanisms named Meshing 1, Meshing 2 and Meshing 3.

All three methods merge neighboring triangles into quadrilaterals. Triangle merging process can sometimes leave unpaired triangles and consequently, small holes in places of such triangles may appear.

Each meshing method can be preceded by the corresponding type of the integrated decimation process.

- Meshing 1 is an algorithm related to the Angle controlled decimation. It allows the formation of small holes in the structure. The existence of such holes reduces number of plates, but also decreases the accuracy of the resulting model.
- Meshing 2 is an algorithm related to the Angle controlled decimation. It does not allow the existence of any hole in the structure. In order to achieve this, an additional division of plates is initiated. Resulting model is completely closed, but with a larger number of plates.
- Meshing 3 is an algorithm related to Deviation controlled decimation. It does not allow the existence of holes in the structure and implements the additional division of plates. During the division, inserted points are shifted to the original model, thereby further reducing the model's deviation.

The Quad-based model, shown in the Fig. 12, is created as a result of the meshing mechanism (Meshing 3) applied on the decimated model in Fig. 11.

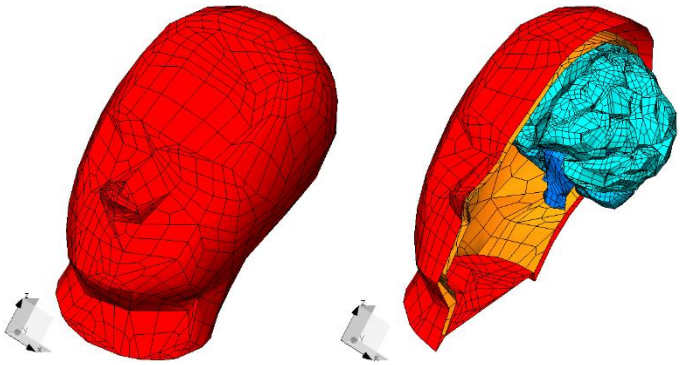

Fig. 12. Meshed model of a head with a brain (made of 11329 plates), full model (left) and model with hidden outer parts (right).

i) Export to .stl format

After each change of the STL project, the project can be saved in .stl format under the same or another name. This process preserves the structure of the model, as well as the domain characteristics (if previously defined).

IWP project geometry can also be exported to .stl format (binary or ASCII). In this process each quadrilateral plate from the original WIPL-D model is subdivided into two triangles. The division is performed along the shorter diagonal of the quadrilateral plate. As STL Editor does not need the information about triangle normals, fields of .stl file used to store this information are used for domain specification.

9. Accelerated Topological Analysis

Topological analysis has never been considered in the past as time consuming operation. However, by increasing the complexity of the problem to tens/hundreds of thousands of patches, this part of simulation time became significant.

In the new version (v17) the topological analysis of the structure is accelerated up to three times.

Example: Let us consider array of printed dipole antennas, the inset of which is shown in Fig. 13. Three sizes of the array are simulated: 28 x 28, 36 x 36, and 44 x 44. One symmetry plane is used to facilitate analysis. In addition to accelerated topological analysis in this example two other novelties are exploited: a) matrix symmetry, and b) accelerated GPU matrix fill. Fig. 14 shows radiation pattern of array consisted of 44 x 44 elements. Table 9 shows number of unknowns and simulations times in v16 and v17 for all three array sizes.

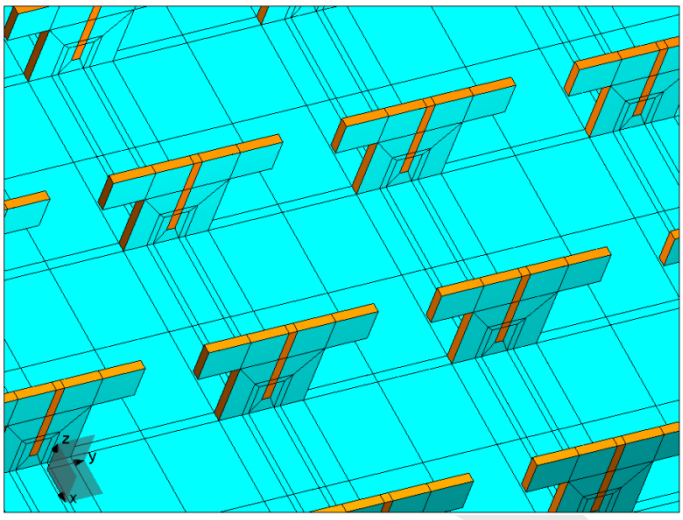

Fig. 13. Inset of meshed model of a printed dipole array.

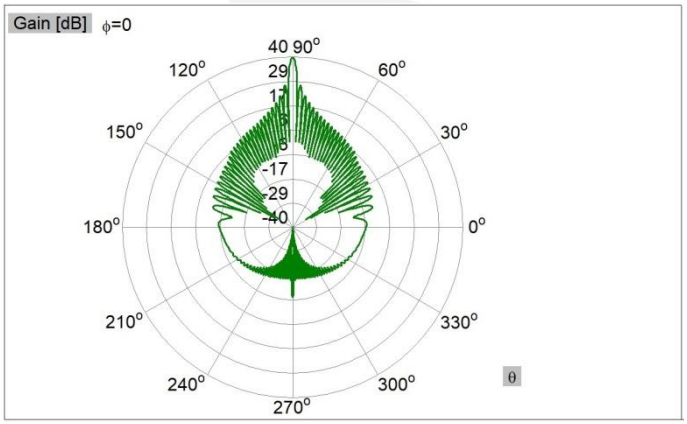

Fig. 14. Radiation pattern of array of 44 x 44 printed dipoles

Table 9. Number of unknowns and total simulation times for three sizes of a printed dipole array, obtained in v16/v17.

Array size	28 x 28	36 x 36	44 x 44
Number of Unknowns	163,296	271,098	405,086
Simulation time v16	4h 15’	11h 54’	> 1 day
Simulation time v17	1h 4’	3h 11’	7h 28.5’

It is seen from the table that in the new version (v17) the simulation time is reduced ~4 times. In particular, in the case of array of 44 x 44 printed dipoles, topological analysis time is reduced from 30’ to 10’.

10. Easy handling of complex structures

By increasing the complexity of the problem to tens/hundreds of thousands of elements, the manipulation of such structures (including Move, Rotate, Scale, and Copy) became limited and demanding in the previous version (V16).

In order to facilitate inspection of such structures two new options are developed in the new version (v17): “Hide Selected Plates” and “Select Layer”, working in the same way as in STL editor (see Section 8).

Limits of “Manipulations” are extended few times. In particular opening and saving the projects is significantly accelerated.