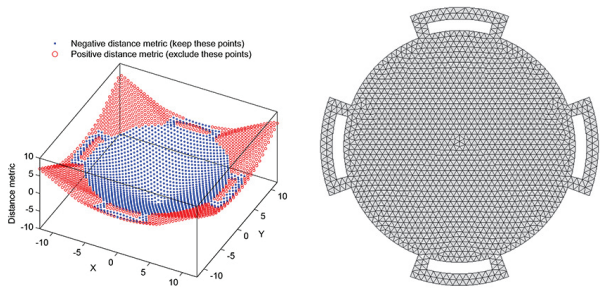
ELECTROSTATIC FINITE-ELEMENT CODE TO STUDY GEOMETRICAL NONLINEAR EFFECTS OF BPMS IN 2D

A. A. Nosych, U. Iriso, ALBA-CELLS, Barcelona, Spain
J. Olle, UAB, Barcelona, Spain

Abstract

We have developed a 2D finite element-based software for Matlab to study non-resonant effects in BPMs of arbitrary geometry, in particular the geometric nonlinearities. The developed code called *BpmLab* utilizes an open-source tetrahedral mesh generator *DistMesh*, combined with a short implementation of FEM with linear basis functions to find the electrostatic field distribution for boundary electric potential excitation. The BPM response as a function of beam position is calculated in a single simulation for all beam positions using the potential ratios, according to the Green's reciprocity theorem. The code offers ways to correct the geometrical nonlinear distortion, either by polynomials or by direct inversion of the electrode signals through numerical optimization. This work is an overview of the *BpmLab* capabilities to date, including its extensive benchmarking and validation against other methods.



(a) Modeling the BPM geometry by metric functions prior to triangular meshing.

(b) Optimized equilateral Delaunay mesh of the BPM's vacuum interior, generated by *DistMesh*.

Figure 1: Modeling the pilot-BPM in *BpmLab* by combining a circle with arcs as functions of radii and internal/orientation angles.

INTRODUCTION

At the initial design stage of any beam position monitor (BPM) lies its numerical characterization and optimization to fit the machine requirements. The common nowadays tools for electromagnetic (EM) simulations are usually an overkill for BPM simulations: they are either expensive (CST, Ansys, GdfidL), or free but complex (ACE3P, POISSON), requiring users to possess specific knowledge of geometrical modeling in 2D/3D and the tool itself. However, BPMs used in accelerators with ultra-relativistic beams are usually designed to be resonance-free and their position characteristic to be independent from the accelerator's operating frequency. Hence, the numerical characterization of a BPM does not need to be complicated and 2D approximations can be made to simulate the behavior of most BPM types and provide their optimization.

At ALBA, characterization of storage ring and booster BPMs has previously been done with the 2D boundary element method (BEM) [1, 2], and lately cross-checked with ACE3P [3]. However, though sufficient for ALBA's needs, the BEM code was limited to only two simple BPM geometries, while using the 3D time domain solver of ACE3P is rather complicated for such tasks.

The electrostatic (ES) approach to numerical BPM characterization is not new and has been used for over a decade in various accelerator laboratories, e.g. SLAC [4], CESR [5] and FNPL [6]. However, together with post-processing, this approach usually requires extra effort of additional software to get the final results.

Matlab is, perhaps, the most common tool for mathematics and data analysis in the accelerator community, with its mighty Middle Layer [7] used in the control systems of a number of accelerators, including ALBA. Thanks to a vast amount of built-in and open-source code solutions offered by Matlab, we have developed *BpmLab*: a simplistic yet powerful code for electrostatic analysis of BPMs of arbitrary geometry and free electrode arrangement in 2D.

METHODOLOGY

Some functionality, mentioned in this work, already exists in Matlab to some extent as parts of paid toolboxes. Our intention, however, is to create a free and easy to use tool, while bringing some novelty to the table.

FEM to Solve Laplace's Equation

In order to solve Poisson's equation we have used a short numerical implementation of Laplace's equation solver in 2D with mixed (Dirichlet + Neumann) boundary conditions for unstructured grids with linear triangular or quadrilateral elements [8]. Here, the problem is solved employing the finite element method (FEM) using the standard Galerkin discretization scheme. The FEM solver calculates the electrostatic potential value in each mesh node based on the boundary conditions and any volume forces, if they exist.

Meshing

For meshing we are using *DistMesh* [9], which is a simple Matlab code for generating constrained equilateral Delaunay meshes with node coordinates optimization by a force-based smoothing procedure. Here the geometry specification is done by *signed distance functions*, which give the shortest distance from every node to the boundary of the domain. The sign is negative inside the region and positive outside,

ISBN 978-3-95450-176-2

Fig. 1(a), and tells the algorithm which nodes to exclude from the final mesh, Fig. 1(b).

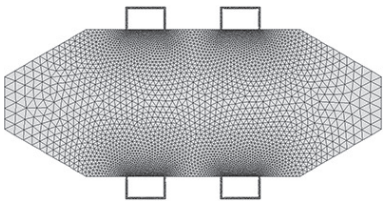


Figure 2: Meshed model of the ALBA storage ring BPM, whose optimized non-uniform Delaunay mesh consists of 8409 triangles on 4558 nodes.

Combining the FEM electrostatic solver with *DistMesh* opens plenty of possibilities for 2D EM modeling, in particular useful for beam instrumentation engineering.

For the moment, we are omitting the Neumann boundary conditions and quadrilateral mesh elements, but focus on fine-tuning the triangular mesh with Dirichlet-type boundaries. We have modified the FEM and *DistMesh* codes and improved their capabilities, e.g. by adding the manual mesh density control in fine geometrical areas (e.g. around the electrodes of ALBA BPM in Fig. 2), the boundary node search for potentials assignment, and the ability to accept any function f as the source of electrostatic field (sometimes also called *volume force* or *space charge*).

Convergence

In order to test the basic reliability of the FEM solver we check convergence of several parameters as functions of mesh edge size. For analytic shapes the error norms L_1 (L_2) are calculated by taking length of vectors whose components are the difference (squared difference) between analytic and FEM solutions in the mesh nodes.

Additionally, a quantity proportional to the system's stored electrostatic energy is calculated, which matches the theoretical result for analytical shapes, and estimates convergence for more complex geometries:

$$\varepsilon = \frac{1}{2}\epsilon_0 \int_V E^2 dV \approx \frac{1}{2}\sum_t {E_t}^2 A_t \text{ [arb. units]} \quad (1)$$

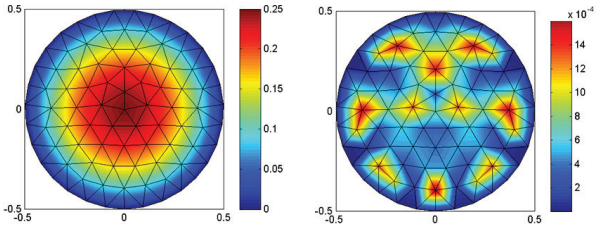
where $\epsilon_0 = 1$ and E^2 is the squared norm of electric field. In Eq. (1) the label t is used to indicate the triangular element we are on and A_t corresponds to its area.

Consider solving the Poisson's equation $-\nabla^2\phi = f$ for each of the two following analytic unit-sized shapes with grounded Dirichlet boundary conditions, which have known analytic solutions:

1) Circle S_1 with radius $R = 0.5$:

$$\begin{cases} \phi_{xx} + \phi_{yy} = -4, & x^2 + y^2 < R^2 \\ \phi = 0, & x^2 + y^2 = R^2 \end{cases} \quad (2)$$

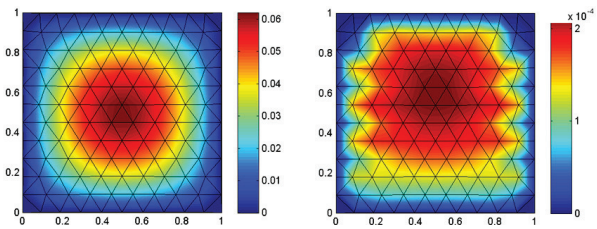
Solution: $\phi(x,y) = R^2 - x^2 - y^2$.



(a) FEM solution on a circle meshed with 145 triangles on 89 nodes.

(b) Difference between FEM and the exact solution, Eq. (2).

Figure 3: Comparison between the FEM and the exact solution of the Poisson's equation for the unit circle shape.



(a) FEM solution on a square meshed with 211 triangles on 128 nodes.

(b) Difference between FEM and the exact solution, Eq. (3).

Figure 4: Comparison between the FEM and the exact solution of the Poisson's equation for the unit square shape.

2) Square S_2 with side length $a = 1$:

$$\begin{cases} -\nabla^2\phi = 2x(a-x) + 2y(a-y), & (x,y) \in S_2 \backslash \Omega S_2 \\ \phi = 0, & (x,y) \in \Omega S_2 \end{cases} \quad (3)$$

Solution: $\phi(x,y) = xy(a-x)(a-y)$.

The exact solutions were compared against the ones obtained with the FEM solver achieving excellent accuracy of under 1 mV even for scarce mesh, Figs. 3 and 4.

Plotting the L_1 and L_2 norms as functions of the decreasing mesh size (thus, the increasing number of nodes), reveals an expected converging trend for norms of S_1 and S_2, Fig. 5, and an interesting behavior of the norms of S_2 which appear smoother. This is because a square is much better approximated by triangles than a circle.

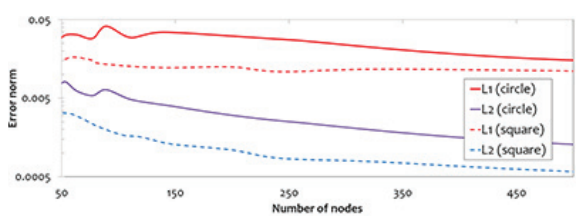


Figure 5: Error norms L_1 and L_2 for the circle and square as functions of the number of nodes.

ISBN 978-3-95450-176-2

SIMULATIONS AND BENCHMARKING

Green's Reciprocity Theorem

Instead of simulating a beam sweep for multiple beam positions, as common with 3D time domain EM solvers in application to BPM modeling, the ES approach offers a similar outcome by a much faster procedure offered by the Green's reciprocity theorem (GRT): obtaining the electrode signals for all beam positions in a single calculation.

In the specific case that we are interested in, GRT states that the charge induced on an electrode surface q_e due to a test charge q at (x_0, y_0) position is proportional to the potential ϕ at that same position when the test charge is absent and the electrode is set to a potential V_0, see [5]:

$$q_e V_0 = -q\phi(x_0, y_0) \tag{4}$$

Therefore, $\phi(x_0, y_0)$ is the solution to the problem of calculating the signal of an electrode, up to a multiplicative constant, as a function of the charge location. This constant, q_e in Eq. (4), is of no importance because we later compute the normalized ratio of potentials, which is equivalent to the normalized ratio of electrode signals. The problem comes down to computing $\phi_i(x, y)$ for each electrode $i = 1..4$ for every mesh node inside the chamber.

We test and validate the developed codes on a 2D model of a showcase BPM labeled *pilot-BPM*. It has 1 mm thick electrodes spanning $\alpha = 30^o$ each, mounted flush in a $0°/90°$ orthogonal arrangement in the circular vacuum chamber of radius $R = 10$ mm and gap of 1 mm.

In practice, the application of GRT comes down to exciting one electrode with non-zero voltage and grounding the others, including the vacuum chamber. The resulting potential $\phi_1(x, y)$ and mesh is then rotated/mirrored to get $\phi_{2,3,4}(x, y)$, as illustrated with the pilot-BPM in Fig. 6. Potential values beyond the mesh nodes are found by linear interpolation which is sufficient due to working with linear finite elements.

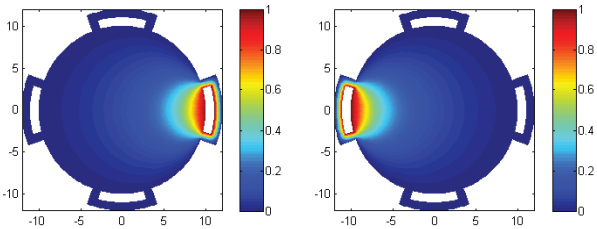
BpmLab offers several standard *difference over sum* (DOS) treatments of BPM signals, which can be applied to a given BPM geometry according to its electrode arrangement.

In particular, the normalized "raw" position characteristic for the circular pilot-BPM is calculated conventionally as

$$x_{\text{raw}} = \frac{\phi_1 - \phi_2}{\phi_1 + \phi_2}, \quad y_{\text{raw}} = \frac{\phi_3 - \phi_4}{\phi_3 + \phi_4} \tag{5}$$

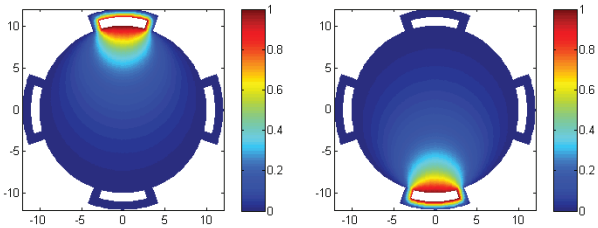
Combining together the potentials $\phi_i(x, y)$ for the pilot-BPM with Eq. (5), the normalized horizontal and vertical beam positions in the locations of mesh nodes are obtained, Fig. 7(a). From here they can be interpolated on a rectangular grid and calibrated to mm by scaling factors $k_{x,y}$ which, for circular beam pipes, can be simply approximated by $k_{x,y} = R/2$, see [10].

Computing the 29×29-point beam position map (±7mm, step of 0.5 mm) for the pilot-BPM results in a typical "pincushion" characteristic map, Fig. 7(b).



(a) Original calculation of $\phi_1(x, y)$ by exciting the right electrode to 1 volt.

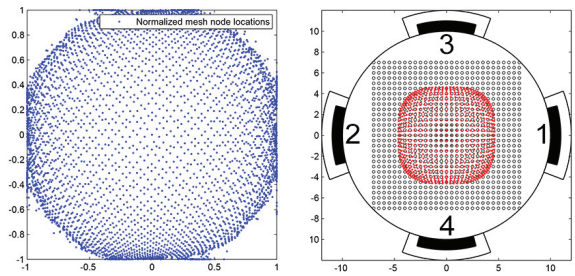
(b) Rotating $\phi_1(x, y)$ by 180° to obtain $\phi_2(x, y)$.



(c) Rotating $\phi_1(x, y)$ by 90° to obtain $\phi_3(x, y)$.

(d) Rotating $\phi_1(x, y)$ by 270° to obtain $\phi_4(x, y)$.

Figure 6: One electrostatic solution rotated 3 times to emulate 3 other excitations.



(a) Normalized DOS ratios of $\phi_{1..4}$ calculated in mesh nodes.

(b) Interpolated and calibrated beam position map.

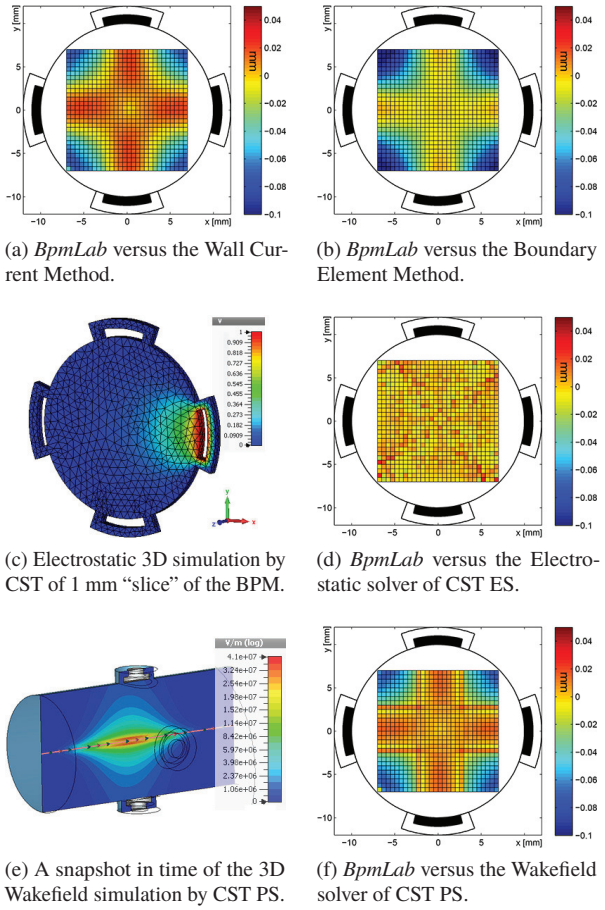
Figure 7: Simulating BPM response for the pilot-BPM: blue dots are the normalized DOS values in node locations, black dots are the true beam positions, and red ones form the nonlinear response map.

Validation Against Other Methods

It is important to validate the results obtained by *BpmLab* with the results obtained by other available methods and tools. The pilot-BPM geometry was simulated by each of the following methods, and differences between position maps, treated by Eq. (5), are shown.

a) ***BpmLab* versus the Wall Current Method (WCM).** For BPMs mounted in circular vacuum chambers the sensitivity function can be estimated analytically by integrating the wall current distribution induced on an electrode due to a line-charge as a function of (x_0, y_0, α, R), [10]. The difference between the beam position maps calculated by *BpmLab* and WCM for the pilot-BPM is below 30 μm in most of the map region except its corners, Fig. 8 (a).

ISBN 978-3-95450-176-2



(a) *BpmLab* versus the Wall Current Method.

(b) *BpmLab* versus the Boundary Element Method.

(c) Electrostatic 3D simulation by CST of 1 mm "slice" of the BPM.

(d) *BpmLab* versus the Electrostatic solver of CST ES.

(e) A snapshot in time of the 3D Wakefield simulation by CST PS.

(f) *BpmLab* versus the Wakefield solver of CST PS.

Figure 8: Differences between beam position maps of the pilot-BPM, calculated by *BpmLab* and other methods.

b) ***BpmLab* versus the Boundary Element Method (BEM).** BEM is based on the numerical solution of the 2D electrostatic problem of finding the induced charge on the *boundary* of the domain containing some charge $\rho(x_0, y_0)$ that represents the relativistic beam. The BEM solver available at ALBA [2] shows mainly a 20 μm difference between the response map calculated by *BpmLab*, Fig. 8 (b).

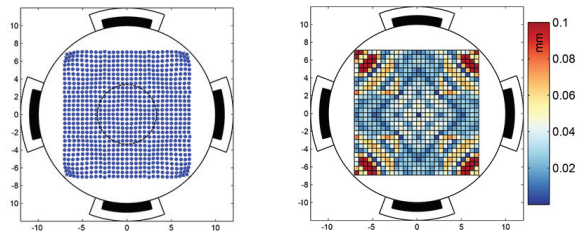
c) *BpmLab* versus the CST Electrostatic solver. Potential excitation of the right electrode to 1 volt is simulated by the Electrostatic solver of CST in Fig. 8(c). After post-processing comparing the maps calculated with CST ES and *BpmLab* show that they essentially overlap with under 10 μm difference, Fig. 8(d).

d) *BpmLab* versus the CST Wakefield solver. Finally, the pilot-BPM was simulated and mapped with the time domain wakefield solver of CST Particle Studio, Fig. 8(e). The 3D model consisted of 200k hexahedral mesh cells; the beam scan was modeled by a single Gaussian relativistic bunch (σ_τ = 5 mm, $Q = 10^{-8}$ C) traveling for 500 ps (80 mm) of wake length. The resulting difference with the response map calculated by *BpmLab* is mainly below 30 μm, Fig. 8(f).

NONLINEARITY CORRECTIONS

Nonlinear distortion of the BPM response can be an issue, especially when large beam offsets are foreseen. Correction of such nonlinearities is integrated into *BpmLab* through:

1) calculation of polynomial coefficients of a non linear fit of two normalized quantities, x_{raw}, y_{raw}, derived from potentials $\phi_{1..4}$. These polynomials allow accurate real-time reconstruction of the transverse beam position taking into account the coupling between electrodes [10, 11]. Low-power polynomials are often sufficient for good precision: Fig. 9 shows the result of applying a 9$^{\text{th}}$ order 2D polynomial to correct the DOS distortion of the pilot-BPM in Fig. 7(b).



(a) Corrected map.

(b) Error of correction in mm.

Figure 9: Response map of the pilot-BPM, corrected by the 9$^{\text{th}}$ order 2D polynomial.

2) direct inversion of the electrode voltages to corresponding beam position using the model behavior of a certain BPM. This procedure is intended for offline post-processing through iterative numerical optimization [5], [11], [12] and promises much more precision over polynomial fits.

This algorithm is implemented in *BpmLab* for an arbitrary 4-electrode BPM geometry. The BPM characteristic is modeled by a set of calculated potentials $\varphi_{i,i=1..4}$ on a non-uniformly distributed array of triangular mesh nodes.

Briefly, to invert a set of voltages $V_{i,i=1..4}$ to the unique beam position inducing them, the following target function is minimized

$$f(x,y) = f_{\text{h}}^2(x,y) + f_{\text{v}}^2(x,y) \rightarrow \min\,, \qquad (6)$$

where

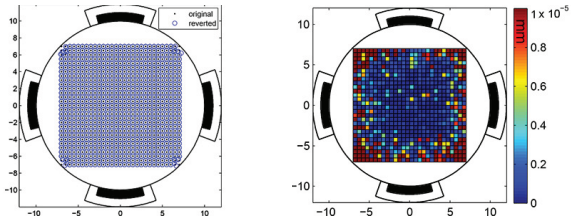
$$f_{\text{h}}(x,y) = \frac{\Delta_{\text{h}}\phi(x,y)}{\Sigma_{\text{h}}\phi(x,y)} - \frac{\Delta_{\text{h}}V}{\Sigma_{\text{h}}V} \qquad (7)$$

and a similar expression for $f_{\text{v}}(x,y)$.

To minimize the target function (6) the *gradient descent method* (GDM) is used, which presupposes that the gradient of the function can be computed numerically or analytically. The DOS ratios of ϕ and V in Eq. (7) follow the particular DOS convention, e.g. Eq. 5.

The search starts at a certain starting point (x_0, y_0) and, as many times as needed, moves from (x_i, y_i) to (x_{i+1}, y_{i+1}) by minimizing along the line in the direction of the local downhill gradient, expressed analytically:

ISBN 978-3-95450-176-2



(a) Corrected map. (b) Error of correction.

Figure 10: Correction based on voltage inversion through optimization. Here the self-generated BPM signals are used for backward convergence test of the FEM method.

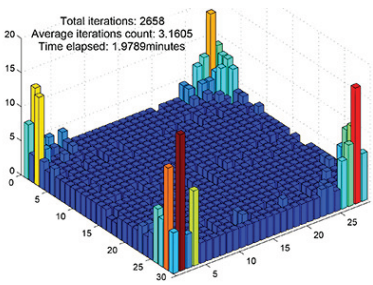


Figure 11: Amount of iterations (represented in bar height and color) it took to invert 4×841 sets of self-generated signals back to 841 beam positions.

$$\nabla f(x_i, y_i) = \left[\frac{\partial f(x_i, y_i)}{\partial x}; \frac{\partial f(x_i, y_i)}{\partial y} \right]. \qquad (8)$$

The starting point is found by yet another minor optimization routine: *the nearest node search*, which looks for the node for which the normalized $(x_{\text{raw}}, y_{\text{raw}})$ of the potentials resemble the corresponding ratio of the voltages.

Assuming that ϕ_i are piece-wise planar functions over the triangulated region, the partial derivatives of $\partial\phi_i$ implied in (8) are found numerically in every node as first order approximations.

The inversion procedure was tested for backward convergence on the FEM model of the pilot-BPM using self-generated signals on a square grid of beam positions. It has shown excellent beam position recovery, Fig. 10(a), with machine precision, Fig. 10(b). The inversion process took under 2 minutes for 29×29 map points taking 3 iterations on average to reach the optimum for each position, Fig. 11.

We have also made an attempt to restore the beam position map based on voltages, generated by the CST Wakefield solver for same map parameters, which resulted in achieving $< 50\mu$m accuracy across most part of the map, Figs. 12.

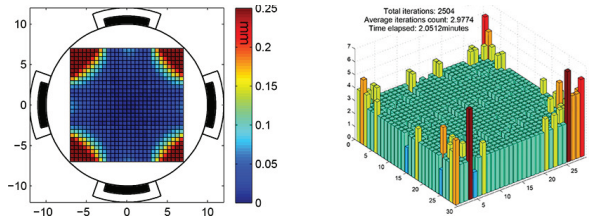
ACKNOWLEDGMENTS

The authors would like to thank M. Wendt and G. Rehm for inspiration and time spent in fruitful discussions.

CONCLUSION

We have created a functional tool to ease the process of BPM modeling for the Matlab platform. Performing a single electrostatic simulation on a modern desktop PC takes negligible amount of time for a meshed BPM model with some 10k nodes, so characterizing BPMs becomes a fast seamless task. The code offers ways to correct for the nonlinear geometrical distortion effect via polynomials, or by voltage inversion through numerical optimization, naturally using the mesh properties.

BpmLab has been extensively benchmarked against a number of other methods and commercial tools. It has shown to be computationally quick and easily adjustable to any BPM geometry, assuring its precise meshing and reliable results.



(a) Corrected map. (b) Inversion iterations.

Figure 12: Correction based on voltage inversion, performed on voltages simulated with the Wakefield solver of CST PS.

REFERENCES

[1] A. Stella, "Analysis of the Dafne beam position monitor with a boundary element method", *Dafne Tech. Note*, **CD-10** (1997).

[2] A. Olmos et al., "Matlab code for BPM button geometry computation", *Proc. DIPAC-2007*, pp. 186-188 (2007).

[3] A. A. Nosych et al, "Overview of the geometrical non-linear effects of button BPMs and methodology for their efficient suppression", *Proc. IBIC'14* (2014).

[4] S. R. Smith, "Beam position monitor engineering", **SLAC-PUB-7244** (1996).

[5] R. W. Helms, G. H. Hoffstaetter, "Orbit and optics improvement by evaluating the nonlinear BPM response in the Cornell Electron Storage Ring", *Phys. Rev. ST AB*, vol. **8**, issue **6** (2005).

[6] P. Piot, "Evaluation and correction of nonlinear effects in FNPL BPM", *FERMI Note* (2005).

[7] G. Portmann et al., "Middle Layer Software Manual for Accelerator Physics", *LBNL Internal Report*, **LSAP-302** (2005).

[8] J. Alberty, C. Carstensen, S. A. Funken, "Remarks around 50 lines of Matlab: short finite element implementation", *Numerical Algorithms* **20**, 117-137 (1999).

[9] P.-O. Persson, G. Strang, "A Simple Mesh Generator in MATLAB", *SIAM Review*, Volume **46 (2),** pp. 329-345 (2004).

[10] A. A. Nosych, M. Wendt, "Analysis of Geometric Nonlinearities of LHC BPMs by 2D and 3D Electromagnetic Simulations", *PRST-AB* (submitted).

[11] R. Bartolini, J. Rowland, "Geometric nonlinearities of a four buttons BPM", *DLS Int. Rep.* **No. AP-SRREP-0174** (2010).

[12] N. Hubert, B. Beranger, L. S. Nadolski, "BPM data correction at Soleil", *Proceedings of IPAC'14* (2014).

ISBN 978-3-95450-176-2