Averaging and Metropolis Iterations for Positron Emission Tomography

László Szirmay-Kalos, Milán Magdics, Balázs Tóth[1], and Tamás Bükki[2]
[1]:Budapest University of Technology and Economics, [2]: Mediso, Hungary

***Abstract*—Iterative Positron Emission Tomography (PET) reconstruction computes projections between the voxel space and the LOR space, which are mathematically equivalent to the evaluation of multi-dimensional integrals. The dimension of the integration domain can be very high if scattering needs to be compensated. Monte Carlo (MC) quadrature is a straightforward method to approximate high-dimensional integrals. As the numbers of voxels and LORs can be in the order of hundred millions and the projection also depends on the measured object, the quadratures cannot be pre-computed, but Monte Carlo simulation should take place on-the-fly during the iterative reconstruction process. This paper presents modifications of the Maximum Likelihood, Expectation Maximization (ML-EM) iteration scheme to reduce the reconstruction error due to the on-the-fly MC approximations of forward and back projections. If the MC sample locations are the same in every iteration step of the ML-EM scheme, then the approximation error will lead to a modified reconstruction result. However, when random estimates are statistically independent in different iteration steps, then the iteration may either diverge or fluctuate around the solution. Our goal is to increase the accuracy and the stability of the iterative solution while keeping the number of random samples and therefore the reconstruction time low. We first analyze the error behavior of ML-EM iteration with on-the-fly MC projections, then propose two solutions: averaging iteration and Metropolis iteration. Averaging iteration averages forward projection estimates during the iteration sequence. Metropolis iteration rejects those forward projection estimates that would compromise the reconstruction and also guarantees the unbiasedness of the tracer density estimate. We demonstrate that these techniques allow a significant reduction of the required number of samples and thus the reconstruction time. The proposed methods are built into the Teratomo™ system.**

I. Introduction

Tomography reconstruction is the *inverse problem* of particle transport in scattering and absorbing media, which requires the iteration of particle transport calculations and corrective back projections [21]. The inputs of the reconstruction are the measured values in *Lines of Responses* or *LORs*: $\mathbf{y} = (y_1, y_2, \ldots, y_{N_{\text{LOR}}})$, and the *material map* of the examined object, which is typically obtained by a CT scan. The output of the reconstruction method is the *tracer density* function $x(\vec{v})$, which is approximated in a *finite function series* form:

$$x(\vec{v}) = \sum_{V=1}^{N_{\text{voxel}}} x_V b_V(\vec{v}), \tag{1}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_{N_{\text{voxel}}})$ are unknown coefficients and $b_V(\vec{v})$ ($V = 1, \ldots, N_{\text{voxel}}$) are *basis functions* [3], which are typically defined on a *voxel grid*. The correspondence between the coefficients of the tracer density function (voxel values) and the LOR hits is established by the *system sensitivity* $T(\vec{v} \to L)$ defining the probability that a radioactive decay happened in $\vec{v}$ is detected by LOR L.

Forward projection computes the expectation value of the number of hits in each LOR L:

$$\tilde{y}_L = \int_{\mathcal{V}} x(\vec{v}) T(\vec{v} \to L) \mathrm{d}v = \sum_{V=1}^{N_{\text{voxel}}} A_{LV} x_V \tag{2}$$

where $\mathcal{V}$ is the domain of the reconstruction, i.e. the field of view of the tomograph, and A_{LV} is the *System Matrix* (*SM*):

$$A_{LV} = \int_{\mathcal{V}} b_V(\vec{v}) T(\vec{v} \to L) \mathrm{d}v \tag{3}$$

The ML-EM scheme searches tracer density coefficients $x_1, \ldots, x_{N_{voxel}}$ that maximize the probability of measurement results $y_1, \ldots, y_{N_{LOR}}$ by an iterative algorithm started from a uniform tracer density or an initial estimate. Taking into account that the measured hits follow a Poisson distribution, after each forward projection, the ML-EM scheme executes a *back projection* correcting the voxel estimates based on the ratios of measured and computed LOR values [22]:

$$x'_V = x_V \cdot \frac{\sum_L A_{LV} \frac{y_L}{\tilde{y}_L}}{\sum_L A_{LV}}. \tag{4}$$

To compute forward and back projections, we should consider all points where positrons can be generated and all possible particle paths that can lead to an event in LOR L. A particle path can be described by a sequence of particle–matter interaction points, thus potential contribution $T(\vec{v} \to L)$ of annihilation point $\vec{v}$ to LOR L is a high-dimensional integral, so are the expected LOR hits in equation 2 and SM elements in equation 3.

SM elements defining projections can be measured [17] or pre-computed off-line [20], but these methods cannot incorporate patient specific material information. Moreover, in fully-3D reconstruction both the number of LORs and the number of voxels may be several hundred millions, thus the SM is prohibitively large and is not sparse if scattering is also taken into account. To handle the huge SM, it can be *factored* according to physical phenomena like positron range, geometric projection, absorption, scattering in the measured object, scattering in the detectors, etc. [19]. Simpler physical phenomena corresponding to lower-dimensional integrals, like geometric projection or absorption, may be obtained by on-the-fly analytic approximations. However, complex phenomena,

like scattering, are associated with very high-dimensional integrals, for which MC quadrature is a powerful and straightforward tool. MC quadrature evaluates the integrand at discrete samples that should be concentrated where the integrand is large [2]. The statistical error bound of MC quadrature is inversely proportional to the square root of the number of samples and does not grow with the dimension of the integration domain. Furthermore, MC quadrature offers a lot of freedom for the definition of the actual sampling strategy, allowing to take into account the features of the computing hardware. This is particularly important if the algorithm runs on the *Graphics Processing Unit* (GPU), which has massively parallel architecture but favors algorithms having limited number of conditional statements and are of *gathering* type, i.e. computational threads write only their private memory locations. The disadvantage of MC simulation is its high computation cost when accurate results are needed. In this paper we focus on computing projections with on-the-fly MC simulation of the physical model during iterative reconstruction. We show that its computational burden can be significantly reduced by distributing the samples among different steps of an iterative reconstruction.

During the development of reconstruction methods computing projections on-the-fly with MC quadrature, we face the following design decisions:

1) Should we generate different, i.e. statistically independent projections in different iteration steps, or should we use the same approximation? Note that the second alternative would be equivalent to the matrix precomputation from precision point of view, but instead of storing the huge matrix, the seed of the random number generator is re-initialized at each iteration step, so pseudo-random methods will lead to the same approximation.
2) In a single iteration step, we need to execute two projections, a forward projection and a back projection. We can ask again whether the forward projector should use the same SM estimate as the back projector of the same iteration step, or it is better to use two statistically independent estimates obtained possibly with different number of samples.
3) During iterative reconstruction, the projectors are applied many times. Is it possible to use the information of previous projections to improve the accuracy of the current one? Of course, such accuracy improvement is feasible only if it does not require the storage of the huge SM.

This paper provides answers to these questions and is organized as follows. Section II presents a motivating example of a 2D PET that randomly estimates an analytically computable SM and examines the convergence properties of the ML-EM iteration involving different sampling methods. Our goal is the modification of the ML-EM iteration to make it stable and accurate even for low sample numbers and high measurement noise. Our proposed solutions are presented in Sections III-A and III-B. Finally, we discuss and evaluate the new iteration schemes in fully-3D PET reconstruction.

II. Motivation and problem statement

For motivation, we first examine a simple analytical problem describing a 2D PET where a SM of dimensions $N_{\mathrm{LOR}} = 2115$ and $N_{\mathrm{voxel}} = 1024$ is defined as the weighted sum of two Gaussian density functions of the distance between the LOR and the voxel with FWHMs equal to the detector size and to five times the detector size, respectively (Fig. 1). The wider Gaussian may be interpreted as the scattered contribution, while the narrower Gaussian may refer to the direct contribution. The Gaussian of the direct contribution is given 60% weight and the scattered contribution 40%, which reflects the typical ratios in human PETs. The analytical nature and the small size of this example provide us with a ground truth SM and reconstruction result to which different alternatives can be compared. However, we have to emphasize that this example is not realistic in the sense that in practice the system matrix is never known and its huge size prohibits the storage of its elements.

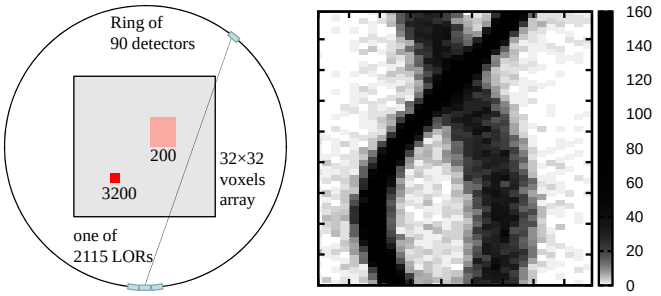


Fig. 1. A simple 2D tomograph model used in the experiments of this section (left). The detector ring contains 90 detector crystals and each of them is of size 2.2 in voxel units and participates in 47 LORs connecting this crystal to crystals being in the opposite half circle, thus the total number of LORs is $90 \times 47/2 = 2115$. The voxel array to be reconstructed is in the middle of the ring and has 32×32 resolution, i.e. 1024 voxels. The ground truth voxel array has two hot squares, one is of 6×6 voxels where each voxel's activity is 200, the other is of 2×2 voxels where each voxel's activity is 3200. Its measured projection involving Poisson noise is shown in sinogram form in the image on the right, where the horizontal and vertical axes correspond to the signed distance of the line from the center and the angle of the line, respectively.

The reference activity is a simple function defined by two hot rectangles of Fig. 1. The measured values are obtained by sampling Poisson distributed random variables setting their means to the product of the SM and the reference activity (right of Fig. 1).

The random system matrices are calculated with 10^5–10^7 samples in total (Fig. 2). Note that estimating $2 \cdot 10^6$ SM elements with 10^5 discrete samples in total means that most of the SM elements get no sample and thus are replaced by zero, making this a very high-variance estimation that can be considered as a stress test for ML-EM reconstruction.

In the first set of experiments we determine the relative L_2 error of the reconstruction process of the *fixed case*, i.e.

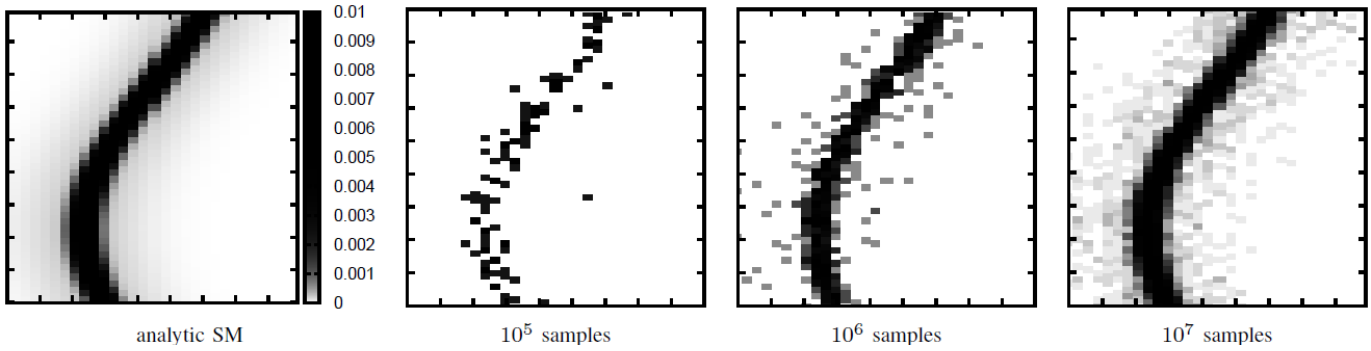


Fig. 2. One column of the SM: sinograms corresponding to voxel $(10, 10)$, which is the right–upper neighbor of the smaller hot region in Fig. 1, when the SM is computed analytically or with 10^5 – 10^7 random samples in total. Note that when only 10^5 random samples are taken to approximate all of the $2 \cdot 10^6$ SM elements, 95% of the elements get no sample and are replaced by zero, while the remaining 5% are approximated by a constant sample weight that is equal to the sum of all SM elements divided by the number of samples. Increasing the number of samples, more than one sample can contribute to a single SM element, thus zeros and small integer multiples of the sample weight show up.

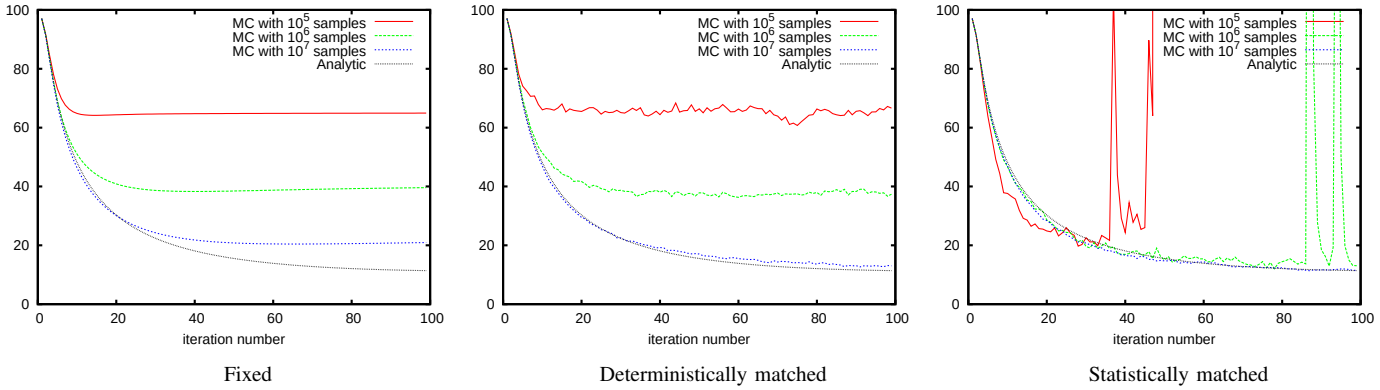


Fig. 3. Relative L_2 error curves obtained with different sampling techniques. Fixed case: SM is the same in every iteration step. Deterministically matched stochastic iteration: SM is re-sampled in each iteration step and the forward projector of an iteration step uses the same SM as its back projector. Statistically matched stochastic iteration: SM is re-sampled for every projection, i.e. the forward projection is statistically independent of the back projection.

when the same SM approximation is used in all iteration steps (left of Fig. 3). These results indicate that working with the same MC estimate during an ML-EM iteration is generally a bad idea unless the estimate has high accuracy. Reconstructing with a modified SM means that we alter the physical model, so the ML-EM iteration converges to a different solution. Unfortunately, accurate MC estimates have very high computation cost.

Another possibility is the independent re-sampling of the SM in each iteration step. Thus, we have the potential to get all information from the SM, although not in a single step, but as the iteration proceeds. The middle of Fig. 3 shows the error curves obtained when we use statistically independent SM estimates in different iteration steps, but in a single step the same SM estimate is applied in forward projection and back projection. This option is called the *deterministically matched stochastic iteration*. The error curves are stable but are just slightly better than those generated with fixed SM.

The right of Fig. 3 depicts the error when not only the system matrices of different iteration steps are statistically independent, but also the forward projector and the back projector. We call it *statistically matched stochastic iteration*. Compared to the deterministically matched method, the statistically matched approach has advantages and disadvantages as well. If the sample number is small, then the statistically matched method is unstable and quickly diverges. If the number of samples is higher, then the statistically matched method is better than the deterministically matched one and gets similar to iterating with the analytic SM.

Fig. 4 shows the reconstruction results after 100 iteration steps for the discussed sampling methods and demonstrate that the fixed and the deterministically matched approaches blur the peaks and edges but are stable, while the statistically matched method behaves similarly to the analytic SM if the sample number is sufficient but may be unstable otherwise, introducing noisy voxels.

Let us explain these observations based on the statistical analysis of the iteration sequence. The ML-EM scheme converges to a single fixed point if the SM is fixed and all columns have non-zero values. However, if a different random projection is used in every step, then iteration schemes do not converge to a single fixed point but either fluctuate or diverge.

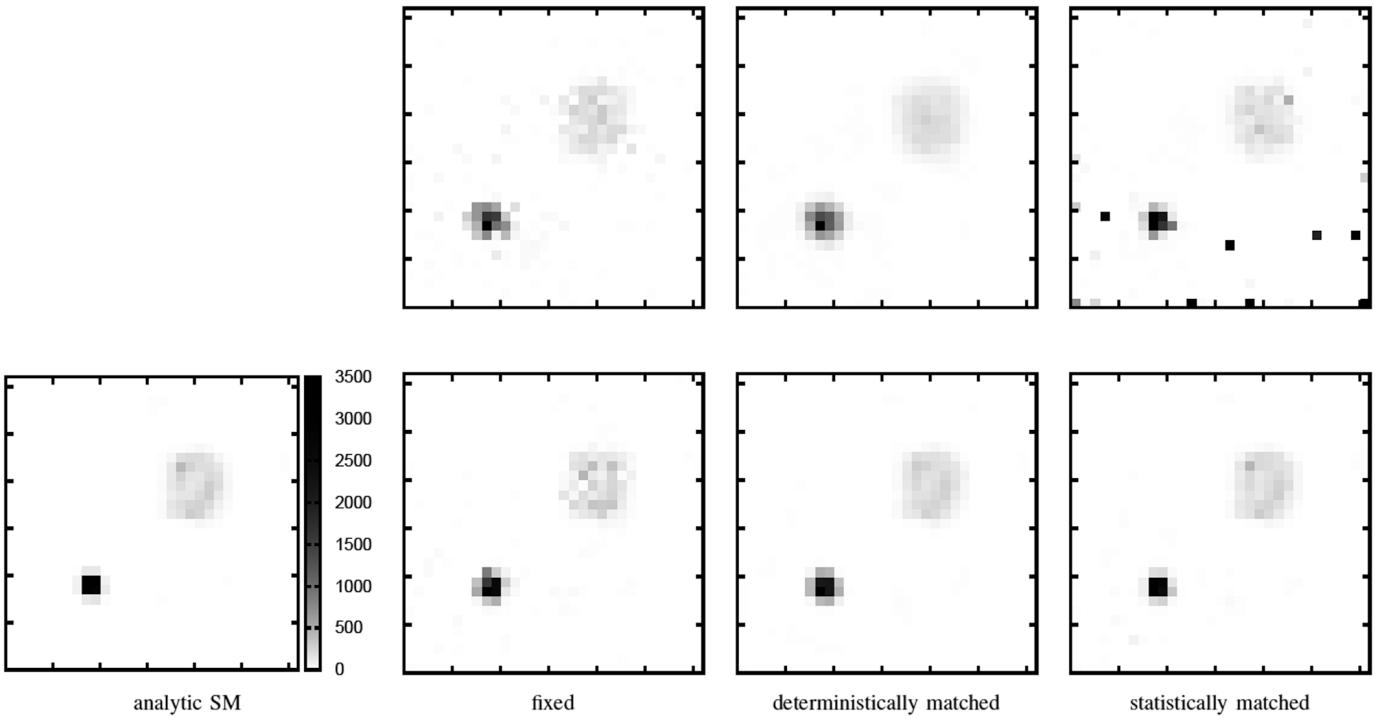


Fig. 4. Reconstructed activity obtained with analytic SM, 10^5 sample projections (upper row) and 10^6 sample projections (lower row) with the discussed sampling techniques. Note that fixed and deterministically matched iteration are stable but fail to quickly converge to higher peaks. Statistically matched iteration, on the other hand, performs similarly to the analytical SM when the sample number is higher but becomes unstable and generates strong voxel noise when the sample number is smaller.

We consider a scheme *accurate* if it fluctuates around the fixed point of the iteration with the analytic SM, and a scheme is said to be *stable* if it does not diverge and the fluctuation remains small.

Accuracy is provided if the expectation of the voxel values after sufficient iterations is similar to the fixed point of iterating with the exact SM, i.e. voxel values have an *unbiased* estimate. An iterated voxel array estimate is the result of the sequence of forward and back projection pairs. A sufficient requirement for the unbiasedness of the voxel estimate would be that all forward and back projectors are unbiased estimates and all projectors are statistically independent, i.e. we can replace the expectation of their products by the products of their expectations. Unfortunately, none of the examined iteration schemes meets the requirement of unbiased back projectors, and fixed and deterministically matched iterations also fail to satisfy the requirement of statistical independence. The forward projector is a linear function of SM elements, thus if SM elements have unbiased estimates, then the computed LOR value $\tilde{y}_L$ has also an unbiased estimate. Unfortunately, this is not true for the back projection, because it is not a linear function of the computed LOR, but depends inversely proportionally to it via ratios $y_L/\tilde{y}_L$ and it also involves the system matrix elements in the denominator as a *sensitivity image* $\sum_L A_{LV}$. The sensitivity image has typically small variance since it involves all SM elements corresponding to a single voxel independently of the actual voxel estimates. However, the ratio of measured and computed LOR values can introduce significant fluctuations unless the forward projection is very accurate.

Stability is related to the error contraction properties of the iteration. Let us assume that the voxel array being the fixed point of the iteration is perturbed in a single voxel V. Error contraction shows how quickly the iteration compensates this perturbation. The perturbation of a single voxel will affect those LORs L during the forward projection where SM elements A_{LV} are significant. In back projection, ratios $y_L/\tilde{y}_L$ are computed, which decrease when the voxel value has increased and increase when the voxel value has decreased. The ML-EM formula will propagate this negative feedback to those voxel values $x_{V'}$ where matrix elements $A_{LV'}$ are significant. The error contraction is high if the originally perturbed voxel V is strongly compensated while other voxel values are left almost unchanged. This condition means that if a voxel has a strong influence on a LOR in the forward projector, then the LOR must also have a strong influence on this voxel in the back projection pair, which is automatically satisfied by the fixed and deterministically matched schemes. In statistically matched iteration, however, the negative feedback is only probable so it may be missing temporarily during the iteration, which may result in larger fluctuations. When the missing deterministic negative feedback is combined with inaccuracy, the corrective steps will fail to fully compensate the accumulated error, and the iteration will be divergent.

III. Modification of the ML-EM scheme

MC quadrature means that a high-dimensional integral is interpreted as the expected value of a multi-dimensional random variable, and then the expected value is approximated by an average of random samples. As computer library functions can return uniformly distributed pseudo-random values, random variables of other distributions are obtained by transforming random variables that are uniformly distributed in the unit domain. Thus, MC quadrature requires the transformation of the integration domain to a unit cube where coordinates correspond to the independently generated uniform random variables. We call this transformed integration domain the *primary sample space* and denote it by $\mathcal{U}$. In a single iteration step we estimate many high-dimensional integrals, one for each LOR in forward projection and one for each voxel in back projection. To prepare for MC estimation in forward projection, the domain of particle paths is transformed to the primary sample space:

$$\tilde{y}_L = \int_{\mathcal{V}} x(\vec{v}) T(\vec{v} \to L) \mathrm{d}v = \int_{\mathcal{U}} \hat{y}_L(\mathbf{u}) \mathrm{d}\mathbf{u} \tag{5}$$

where $\hat{y}(\mathbf{u})$ is the LOR hit estimate associated with the random variable samples in $\mathbf{u}$. The probability density of random variables uniformly distributed in the unit cube is $p(\mathbf{u}) = 1$, thus this integral is the expected value of $\hat{y}_L(\mathbf{u})$, which can be approximated from a single point $\mathbf{u}$ if the fluctuation (variance) of $\hat{y}_L(\mathbf{u})$ is small.

According to the conclusions of the motivating example, the numerical error due to on-the-fly MC estimates and the non-linear dependence of modified voxel values on computed LOR estimates together lead to biased voxel estimates. Let us consider the expectation of the ratio of measured and computed hits, $y_L/\hat{y}_L$. According to the relation of harmonic and arithmetic means, or equivalently to the Jensen's inequality taking into account that $1/\hat{y}_L$ is a convex function, we obtain:

$$E\left[\frac{y_L}{\hat{y}_L(\mathbf{u})}\right] = \int_{\mathcal{U}} \frac{y_L}{\hat{y}_L(\mathbf{u})} \mathrm{d}\mathbf{u} \geq \frac{y_L}{\int_{\mathcal{U}} \hat{y}_L(\mathbf{u}) \mathrm{d}\mathbf{u}} = \frac{y_L}{\tilde{y}_L}. \tag{6}$$

This inequality states that $y_L/\tilde{y}_L$ has a random estimator of positive bias. An intuitive graphical interpretation of this result is shown by Fig. 5. Here we assume that the iteration is already close to the fixed point, so different estimates are around the expected detector hit corresponding to the maximum likelihood. Note that the division in the back projection may amplify forward projection error causing large fluctuations, especially when $\tilde{y}_L$ is close to zero.

We propose two modified iteration schemes to solve this problem, averaging iteration and Metropolis iteration, which are presented in the next sections.

A. Averaging iteration

The bias and the fluctuation of the voxel intensity due to the MC estimate of forward projection can be reduced by making the forward projection more accurate. Exploiting additional samples from previous iteration steps, averaging iteration improves accuracy of the current step without requiring more samples or more processing time. MC estimation in forward projection results in computed LOR hit values $\hat{y}_L$ that fluctuate around their exact value $\tilde{y}_L$. Thus, if MC estimates of subsequent iteration steps use independent random numbers, it is worth averaging the calculated LOR hits obtained in different iteration steps to reduce the scale of the fluctuation.

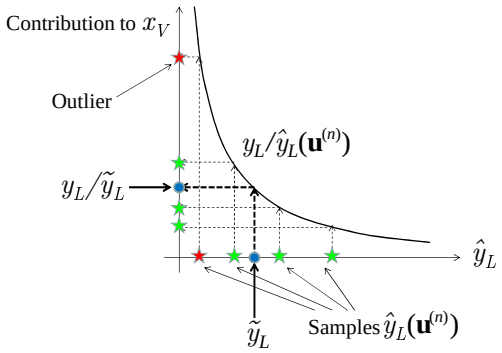


Fig. 5. Expected LOR hit number $\tilde{y}_L$ is approximated by random samples $\hat{y}_L(\mathbf{u}^{(n)})$ in iteration step n, which have mean $\tilde{y}_L$. These random samples are shown on the horizontal axis. Back projection computes ratio $y_L/\hat{y}_L(\mathbf{u}^{(n)})$ to obtain voxel updates, which is a non-linear, convex function, resulting in voxel values that may be much higher than the correct value $y_L/\tilde{y}_L$. These overshooting samples are responsible for a positive bias and occasionally cause a large random increase in the voxel value.

Formally, we obtain the expected LOR hits $\tilde{y}_L^{(n)}$ in iteration step n as the weighted average of the actual MC estimate $\hat{y}_L(\mathbf{u}^{(n)})$ and its previous value $\tilde{y}_L^{(n-1)}$:

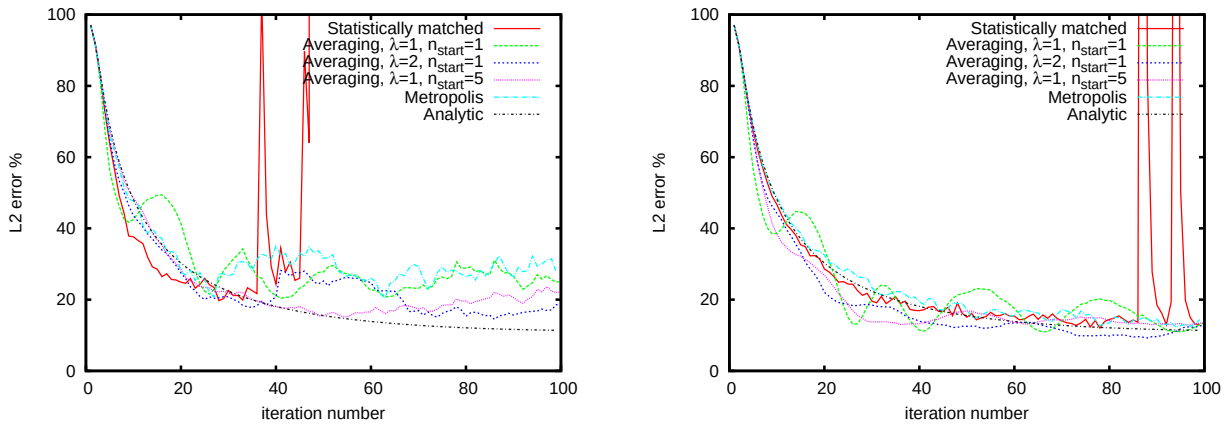
$$\tilde{y}_L^{(n)} = (1 - \tau_n)\, \tilde{y}_L^{(n-1)} + \tau_n \hat{y}_L(\mathbf{u}^{(n)}) \tag{7}$$

where τ_n is the decreasing weight of the estimate obtained in the current iteration step. The weighting scheme can be defined, for example, as $\tau_n = \min(\lambda/n, 1)$, where $\lambda \geq 1$ is a user defined parameter describing how quickly averaging iteration forgets earlier results.

The ML-EM algorithm incorporating averaging in forward projection is as follows:

for $n = 1$ **to** m **do** *// iterations*
  for $L = 1$ **to** N_{LOR} **do** *// Forward project + average*
    $\hat{y}_L =$ Forward Project $\mathbf{x}^{(n-1)}$ with a MC algorithm.
    $\tau_n = \min(\lambda/n, 1)$.
    $\tilde{y}_L^{(n)} = (1 - \tau_n)\, \tilde{y}_L^{(n-1)} + \tau_n \hat{y}_L$.
  endfor
  for $V = 1$ **to** N_{voxel} **do** *// Back project*
    $x_V^{(n)} =$ Back Project $y_L/\tilde{y}_L^{(n)}$ with a MC algorithm.
  endfor
endfor

If we are close to the fixed point and execute m additional iterations with $\tau_m = 1/m$, then averaging iteration is similar to iterating with the average of the system matrices, i.e. with the matrix that is computed using m times more samples. However, when we are farther from the fixed point, LOR estimates



10^5 sample projections 10^6 sample projections

Fig. 6. Relative L_2 error curves of averaging and Metropolis iterations and their comparison to statistically matched iteration. The waves in the error curve of averaging iteration started at the first step with $\lambda = 1$ is eliminated by either starting averaging just at step $n_{\text{start}} = 5$ or setting $\lambda = 2$.

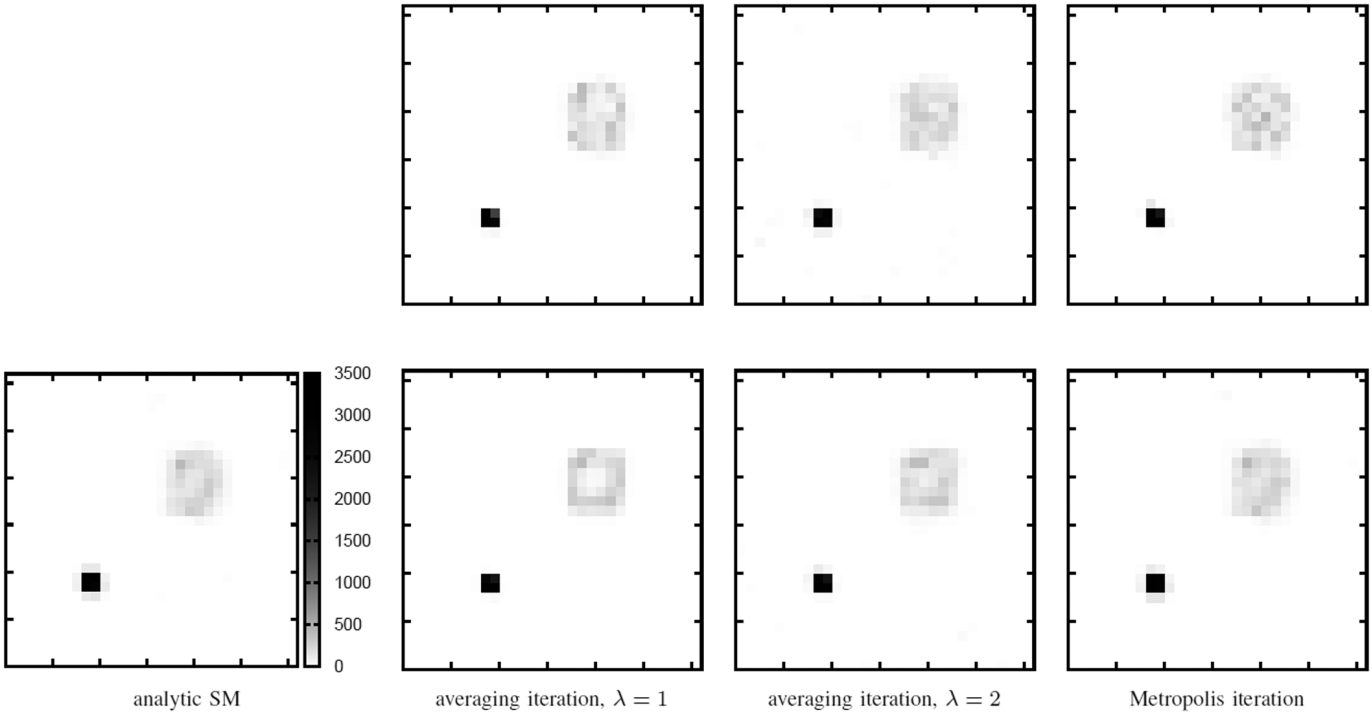


Fig. 7. Reconstructed activity obtained with analytic SM, and with averaging and Metropolis iterations using 10^5 sample projections (upper row) and 10^6 sample projections (lower row).

$\hat{y}_L$ are different not only due to the random fluctuation of the MC sampling but also because of the early evolution of the reconstructed activity $\mathbf{x}^{(n)}$. Averaging iteration reduces random fluctuations, but also slows down the convergence towards the solution having the maximum likelihood especially when there are still significant differences between subsequent iteration steps. This problem can be solved by starting averaging only later in the iteration sequence, or by increasing parameter λ. Note that $\lambda = \infty$ corresponds to statistically matched iteration.

Fig. 6 compares the relative L_2 error curves of averaging iteration using statistically independent forward and back projections and statistically matched iteration for the example of Section II, and Fig. 7 depicts the reconstructions. Note that averaging iteration is stable unlike statistically matched iteration even for small sample numbers. Its wavy L_2 curve is due to the problem that averaging is not fast enough to forget estimates of the first iteration steps, which can be solved by starting averaging at iteration step $n_{\text{start}} = 5$ or by increasing parameter λ from 1 to 2.

B. Metropolis iteration

First, we present Metropolis iteration intuitively, based on the analysis of Fig. 5. The problems of positive bias and the large fluctuations are caused by random samples $\hat{y}_L(\mathbf{u})$ that are much lower than their expected value $\tilde{y}_L$ and result in large overshooting values $y_L/\hat{y}_L(\mathbf{u})$ in the voxel contributions. To attack this problem, these overshooting samples (outliers) are rejected. We suppose that the MC algorithm provides us with a sequence of random *tentative samples* during the iteration, from which real samples are generated by rejecting the outliers and replacing them with the last accepted sample. On the one hand, such replacement would decrease the expectation of the voxel contribution, thus the positive bias can be eliminated. On the other hand, the probability of very large $y_L/\hat{y}_L(\mathbf{u})$ ratios is decreased, so is the probability of fluctuations when these effects are added in different LORs.

A classical MC forward projector obtains samples in the primary sample space with uniform probability density, and transforms these samples as required by the particular algorithm. The added rejection or replacement scheme modifies the uniform probability in the primary sample space. We wish to have a rejection scheme and an associated probability density $p_{\mathrm{Met}}(\mathbf{u})$ that make the updated voxels have unbiased estimates. Let us show that this requirement is met if density $p_{\mathrm{Met}}(\mathbf{u})$ is proportional to forward projection estimate $\hat{y}_L(\mathbf{u})$. The ratio of proportionality is obtained from the requirement that $p_{\mathrm{Met}}(\mathbf{u})$ is a probability density, thus its integral is equal to 1:

$$p_{\mathrm{Met}}(\mathbf{u}) = \frac{\hat{y}_L(\mathbf{u})}{\int\limits_{\mathcal{U}} \hat{y}_L(\mathbf{u})\mathrm{d}\mathbf{u}} = \frac{\hat{y}_L(\mathbf{u})}{\tilde{y}_L}. \tag{8}$$

The expectation of random estimate $y_L/\hat{y}_L(\mathbf{u})$ is then indeed equal to the exact ratio:

$$E_{\mathrm{Met}}\left[\frac{y_L}{\hat{y}_L(\mathbf{u})}\right] = \int\limits_{\mathcal{U}} \frac{y_L}{\hat{y}_L(\mathbf{u})} p_{\mathrm{Met}}(\mathbf{u})\mathrm{d}\mathbf{u} = \frac{y_L}{\tilde{y}_L}. \tag{9}$$

The only remaining task is the elaboration of a rejection scheme that keeps a sample with probability density $p_{\mathrm{Met}}(\mathbf{u}) \propto \hat{y}_L(\mathbf{u})$. Such tasks can be solved with the *Metropolis method* [15]. The sequence of tentative samples $\mathbf{u}^{(n)}$ are uniformly distributed in the primary sample space and are statistically independent. Metropolis sampling establishes a Markov chain $\mathbf{u}_{\mathrm{Met}}^{(n)}$ by randomly rejecting a new tentative element $\mathbf{u}^{(n)}$ based on its contribution $\hat{y}_L(\mathbf{u}^{(n)})$ and on the contribution of the previously accepted sample $\hat{y}_L(\mathbf{u}_{\mathrm{Met}}^{(n-1)})$. The decision uses the *acceptance probability* $a(\mathbf{u}^{(n)})$ that is the ratio of the contributions of the tentative sample and the previously accepted sample.

The state transition probability of the Markov chain is

$$P(\mathbf{u} \to \mathbf{u}') = \min\left(\frac{\hat{y}_L(\mathbf{u}')}{\hat{y}_L(\mathbf{u})}, 1\right). \tag{10}$$

Thus, the ratio of state transition probabilities in two directions between two states is

$$\frac{P(\mathbf{u} \to \mathbf{u}')}{P(\mathbf{u}' \to \mathbf{u})} = \frac{\hat{y}_L(\mathbf{u}')}{\hat{y}_L(\mathbf{u})}. \tag{11}$$

As tentative samples are generated for each primary sample space point associated with a non-zero contribution, the established Markov chain is ergodic, i.e. it has a unique stationary distribution $p_\infty(\mathbf{u}) = \lim p_n(\mathbf{u})$ which is independent of the initial state. The stationary distribution must satisfy the *balance requirement*, i.e. the probability of outflow from a state equals to the probability of inflow, thus

$$\int\limits_{\mathcal{U}} p_\infty(\mathbf{u}) P(\mathbf{u} \to \mathbf{u}')\mathrm{d}\mathbf{u}' = \int\limits_{\mathcal{U}} p_\infty(\mathbf{u}') P(\mathbf{u}' \to \mathbf{u})\mathrm{d}\mathbf{u}'. \tag{12}$$

Using equation 11, it is easy to see that the balance requirement is met when $p_\infty(\mathbf{u}) \propto \hat{y}_L(\mathbf{u})$, and the condition of uniqueness guarantees that the sample density will converge to this distribution.

The ML-EM algorithm incorporating Metropolis sampling in forward projection is as follows:

for $n = 1$ **to** m **do** *// iterations*
  for $L = 1$ **to** $L = N_{\mathrm{LOR}}$ **do** *// Forward project*
    $\hat{y}_L$ = Forward Project $\mathbf{x}^{(n-1)}$ with a MC algorithm.
    $a_L = \min\{\hat{y}_L/\tilde{y}_L^{(n)}, 1\}$. *// acceptance probability*
    Generate random number r in $[0, 1)$.
    if $r < a_L$ **then** $\tilde{y}_L^{(n)} = \hat{y}_L$ *// accept with probability* a_L
    else $\tilde{y}_L^{(n)} = \tilde{y}_L^{(n-1)}$
  endfor
  for $V = 1$ **to** $V = N_{\mathrm{voxel}}$ **do** *// Back project*
    $x_V^{(n)}$ = Back Project $y_L/\tilde{y}_L^{(n)}$ with a MC algorithm.
  endfor
endfor

In the stationary case, the Markov process generates samples with a density proportional to $\hat{y}_L(\mathbf{u})$, but early samples may be drawn from a different distribution. This may result in a *start-up bias*, which is typically handled by ignoring the first few samples corresponding to the burn-in period while the process is not stationary yet. However, we do not have to ignore early samples because of the following two reasons. As our method generates tentative samples independently of the current sample, the perturbation is as large as the whole primary sample space, thus the start-up bias disappears quickly. On the other hand, instead of computing just a single integral, we execute an iteration where each step requires its own projection integrals. Even if some error is made early in the iteration due to the start-up bias when the activity is only roughly estimated anyway, the error will be corrected by later iteration steps when the start-up bias already vanishes.

The relative L_2 error curves of Metropolis iteration are also included in Fig. 6 and its reconstruction result is compared to averaging iteration in Fig. 7. We can observe that the Metropolis method has higher fluctuation than averaging iteration but does not introduce waves in the error curves. The superior stability of averaging iteration is due to the fact that it exploits the samples of all previous iteration steps when the forward projection is estimated while Metropolis iteration effectively combines just the last iteration steps. However, this is also an advantage since the convergence of Metropolis is not slowed

down by the effect of earlier samples, and therefore it does not require additional, volume dependent parameters like λ or start of averaging n_{start}. We also note that averaging iteration not only reduces the variance of computed LOR values, but also suppresses the statistical dependence of forward and back projections since computed LOR values will depend on the samples of all iteration steps. Thus, averaging iteration can also improve deterministically matched iteration schemes. However, this is not true for Metropolis iteration, which preserves forward projection values with higher probability. Therefore, Metropolis sampling is recommended only for statistically matched approaches.

Method	30% L_2 error	20% L_2 error
Fixed	80	300
Deterministically matched	80	290
Statistically matched	17	37
Averaging ($\lambda = 2$)	2	11
Metropolis	6	19

TABLE I
TOTAL NUMBER OF SAMPLES IN MILLIONS NEEDED TO TAKE THE L_2 ERROR BELOW 20% AND 30%, RESPECTIVELY.

The relative performance of different sampling techniques can be characterized by counting the total number of samples — i.e. the product of the number of samples per iteration and the number of iterations — needed to reduce and keep the error below a given threshold (note that stochastic sampling has fluctuating error curve, so we need to find the number of samples that guarantees that the maximum of the fluctuation is less than the threshold). Table I shows a parameter study performed by reconstructing the data of Section II with sample numbers per iteration in the range of 10^5–10^7 and finding the minimum of the product of the sample number and the number of iterations. As the reconstruction time is proportional to the total number of samples, this table shows the relative speed of different methods. For example, averaging iteration is 25–40 times faster than fixed iteration, which may be considered as a classical method, and 3–8 times faster than statistically matched stochastic iteration. Metropolis iteration, on the other hand, is 13 times faster than fixed iteration and 2–3 times faster than statistically matched iteration.

IV. APPLICATION IN PET RECONSTRUCTION

The presented averaging and Metropolis iteration schemes can be included into ML-EM or OSEM algorithms working with on-the-fly MC projections. In practical fully-3D PETs, the system matrix is huge and is not available, the true projection model is defined by physical properties and phenomena that should be simulated by the MC algorithm. In this section we consider different factored phases of a PET reconstruction algorithm, including geometric projection, scattering in the detector, and scattering in the measured object. However, we note that the proposed scheme can also be used with other projection models, including, for example, not factored MC particle transport algorithms [26] or processing time-of-flight data as well.

A. Geometric projections

To demonstrate the application of averaging and Metropolis iterations in geometric projection, we took a LOR-centric, i.e. ray-based forward projection and a voxel-centric back projection, which are particularly suitable for GPU implementation since a computational thread of forward projection computes a LOR while a thread of back projection calculates a voxel, making both of them *output-driven*, also called *gathering* type algorithms [6], [7]. Although forward projection and back projection are based on different sampling strategies and draw samples from different distributions, both of them are unbiased estimates, thus they can form a statistically matched pair.

Forward projection samples are multiple rays [16] or line segments connecting two uniformly distributed points on the two crystals' faces of the LOR. Note that sampling the end points uniformly on the crystal surfaces generates lines that are non-uniformly distributed, thus the contribution of rays should be weighted with a geometric factor [10], [11], [16]. A point on the crystal face is defined in a coordinate system where the origin is a corner of the crystal face and the units on the two axes are equal to the crystal width and height. Such coordinates are in $[0, 1)$ and of uniform distribution, so can be interpreted as coordinates of a primary sample space point. The line integrals of the activity and the attenuation along a ray are evaluated between the two endpoints either analytically or with *ray marching*. For example, Siddon's algorithm [23] can be used for analytical integration if the activity and the material density are represented with piece-wise constant basis functions, or the method of [24] for piece-wise tri-linear basis functions. Ray marching takes constant length steps along the ray and samples the activity and the material density at finite number of sample points. The first sample location is jittered randomly to guarantee unbiased estimate for the line integral. We allocate the same N_{ray} number of rays for each LOR. Assigning unique normalized coordinates to each LOR, the dimension of the primary sample space is $4N_{\text{ray}}N_{\text{LOR}}$ if analytical line integration is executed and $5N_{\text{ray}}N_{\text{LOR}}$ if jittered ray-marching is applied since a ray is defined by its two endpoints, each requiring two coordinates, and jittering requires one additional coordinate to define the location of the first step. However, sharing the normalized coordinates among LORs, the dimension is $4N_{\text{ray}}$ for analytical line integration and $5N_{\text{ray}}$ for jittered ray-marching. We prefer shared samples since they require significantly smaller memory, which is a crucial factor in the GPU.

In back projection, voxel space sample points are obtained for each voxel V with the probability density of $b_V(\vec{x})$. For each voxel sample, sample points on the detector surface are considered. A voxel sample and a detector sample together define a line, which intersects the surface of another detector module, establishing a sample for the LOR. In high resolution reconstruction when voxels are small with respect to the crystals, the number of voxel samples can be as low as one. When detector surface samples are found, we allocate the same number of line samples for each voxel, but only that part of

the detector module is considered from where LORs can be established via this voxel. In the middle of the field-of-view, this means one line sample per LOR, but close to the boundary of the field-of-view, where a voxel is intersected by fewer LORs, LORs are evaluated more accurately. The system matrix elements needed by the ML-EM formula can then be computed from the MC estimates involving the solid angle subtended by the sampled crystals from the voxel sample. If we sample random points just in a reference voxel and translate them with the relative corner locations of each voxel, and detector samples are generated by shifting a 2D grid with a random offset, then the dimension of the primary sample space of the back projection is 5. Note that this method approximates the 5D integral of the line spread functions [4] by random sampling in the 3D space of voxel points and by randomized systematic sampling on the 2D detector surface.

B. Scattering in the detectors

As practical PET detector crystals are not ideal absorbers, gamma photons may go through them without interaction or may get scattered to a different crystal. These phenomena cause blurring and down-scaling of the detected values, which can be calculated as a 4D LOR space filtering [19], [21], [18]. The 4D filter kernel can be expressed as a product of two incident direction dependent 2D filter kernels since the two gamma photons behave independently in this respect. The 2D filter kernel representing scatter in the detectors, photon flying through without detection, absorption, gamma-photon sharing, etc. can be computed off-line with GATE [5]. During GATE simulation, crystals are assumed to be similar. Individual crystal efficiencies are obtained by measurements and are applied as a corrective factor.

In LYSO crystals the average free path length of 511 keV gamma-photons is about 13 mm, while in small animal PETs crystals are packed at about 1 mm distance, thus with non-negligible probability, photons of larger incident angles can be detected farther than the tenth neighbor of the crystal where the photon entered the detector module [8]. Thus, the LOR space filter should cover at least 10×10 crystals on both ends of the LOR, which makes the number of terms to be summed for each LOR greater than 10^4. The prohibitive cost of such large LOR convolution cannot be reduced by filtering in the frequency domain since the 4D filter kernel also depends on the incident directions, and therefore spatially variant in LOR space. To reduce the computation time, the convolution is evaluated by MC estimation taking just N_{det} random offset pairs on the two detectors [12]. According to the concepts of *importance sampling*, random offsets are generated at the two ends of the ray with a probability density of the translation between the gamma photon incidence and absorption. Modifying the two endpoints of a ray needs two 2D translations, i.e. 4 uniformly distributed variables that are transformed to have the probability density of the gamma photon transfer. Thus, if the same random offsets are used in all LORs, the dimension of the primary sample space is $4N_{\mathrm{det}}$. A set of random offsets can be pre-generated off-line and hardwired in the reconstruction program. During reconstruction, each iteration takes a different subset of the pre-generated list according to the concepts of stochastic iteration. Depending on whether the same or different subsets are used in forward projection and back projection of an iteration step, the method is either deterministically or statistically matched.

C. Scattering in the measured object

If we consider photon scattering in the measured object, the path of the photon pair will be a *polyline* containing the emission point somewhere inside one of its line segments. This polyline includes scattering points where one of the photons changed its direction in addition to detector hit points. Considering the *single scattering problem*, the number of scattering points is limited to one. The scattered contribution to a LOR will be the integral of path contributions, which is computed applying Watson's method [25] with the following modifications:

1) Instead of computing the polyline contribution separately for each LOR, the process is decomposed to three phases to allow the reuse of line integrals in multiple LORs. In the first phase, N_{scatter} scattering points are sampled globally, which will be used in the quadratures of all LORs. In the second phase, each detector crystal is connected to each of the scattering points, and along these line segments the line integrals of the total activity, total attenuation due to the photoelectric effect and total out-scattering due to the Compton effect are computed. Absorbtion and scattering cross sections depend on the energy of gamma photons, which is not available yet, thus these line integrals are temporarily computed assuming 511 keV photons. In the third phase, the line segments sharing a scattering point are paired, resulting in N_{scatter} polylines in each LOR. When a polyline is formed, the scattering angle and thus the Compton formula can be evaluated. The total out-scattering and attenuation are corrected according to the ratios of the real photon energy and 511 keV energy [11].
2) Instead of uniform sampling, global scattering points are sampled from a probability density that mimics the scattering cross section, and the contribution is simultaneously divided with this density since this reduces the variance of the MC quadrature according to *importance sampling*.

Note that this three-phase approach significantly reduces the number of line integrals to be computed as it reuses results obtained in the first two phases. As a side effect of reusing samples, the approximation errors in different LORs are correlated, thus the reconstruction will be free of dot noise typical in other MC algorithms.

The estimate of the scattered contribution is added to the direct contribution computed with attenuation at the end of the forward projection. Back projection is simplified and involves geometric effects, attenuation of the measured object and detector phenomena, but not scattering in the measured object, thus the back-projector of this example is not fully matched.

In a single iteration, we need N_{scatter} global scattering points, which can be transformed from $3N_{\text{scatter}}$ uniformly distributed random variables.

The scattered contribution may be computed on lower resolution both in voxel space and LOR space, which improves performance but has the overhead of down-sampling the arrays before integration. We note that the proposed iteration schemes are compatible with other MC scattering computation methods as well. For example, in addition to the single scatter compensation, we can also consider multiple scattering by generating longer paths [11] or by applying re-scaling of the single scatter result with the modification of the scattering cross section [9].

V. Results

The proposed algorithms have been implemented in CUDA, integrated into the Teratomo™ system, and run on NVIDIA GeForce 690 GTX GPUs. We modeled a small animal PET and a human PET to examine the performance of the proposed algorithms.

A. Geometric projections and scattering in the detectors

Geometric projections without and with detector scattering calculation are tested with Mediso's small animal *nanoScan-PET/CT* [14], which has 12 detector modules consisting of 81×39 crystals of size $1.12^2 \times 13$ mm^3. A module's crystal makes a pair with every crystal of five opposite modules. The number of LORs is $N_{\text{LOR}} = 12 \times 5 \times (81 \times 39)^2/2 \approx 3 \cdot 10^8$.

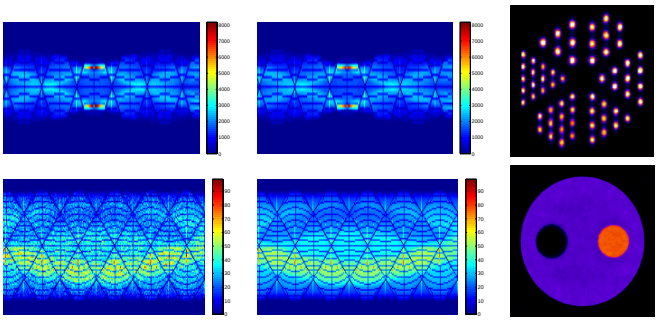


Fig. 8. Validation of the projection model. The Derenzo and the Cylinder are projected with GATE and also with our projector computing scattering in the detector with $N_{\text{det}} = 32$ samples per LOR. The Cylinder phantom contains a hot and a cold smaller cylinders embedded in the large cylinder.

Our MC projector model has been validated by comparing its projections to those obtained with GATE [1] (Fig. 8). Note that the sinograms are similar apart from the random noise that is present both in GATE's and our projections.

To test the efficiency of averaging and Metropolis iterations, first we took an *off-axis point source* of 0.1 MBq activity, placed 2 mm North and 1 mm East from the axis and simulated a 10 sec long measurement with GATE to obtain the input for the reconstruction. To make the simulation consistent with the geometric reconstruction, we set the detectors to very dense material with small free path length. We run four

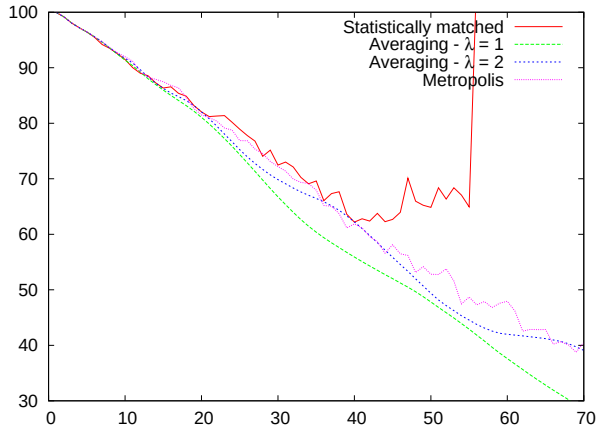


Fig. 9. Relative L_2 error with respect to the number of iterations reconstructing the off-axis point source.

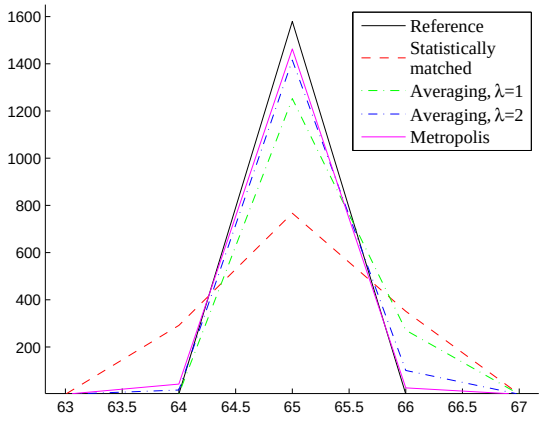


Fig. 10. Profile curves of the point source reconstructions obtained with 100 statistically matched, averaging and Metropolis iterations, respectively. The activity density is in Bqs/mm^3, the unit on the horizontal axis is the edge size of a voxel that is equal to 0.185 mm.

reconstructions of the GATE simulation: averaging iteration with two λ factors, Metropolis iteration, and also statistically matched iteration for comparison. Fig. 9 shows the relative L_2 error curves of the reconstruction of the point source using 0.185 mm^3 voxels and $N_{\text{ray}} = 4$ line samples per LOR. Fig. 10 depicts the line profiles of the reconstructed trilinear activity density. Note that statistically matched iteration exhibits drastic oscillations in the error value and results in a blurred reconstruction, unlike averaging and Metropolis iteration methods. We repeated the statistically matched iteration with $N_{\text{ray}} = 8, 16$ and 24 samples, and compared them to the 4 sample averaging or Metropolis iterations. We concluded that statistically matched iteration gets better than the averaging and Metropolis iterations if it uses more than 16 samples instead of 4. It means that averaging and Metropolis iterations allow 4–5 times faster projections.

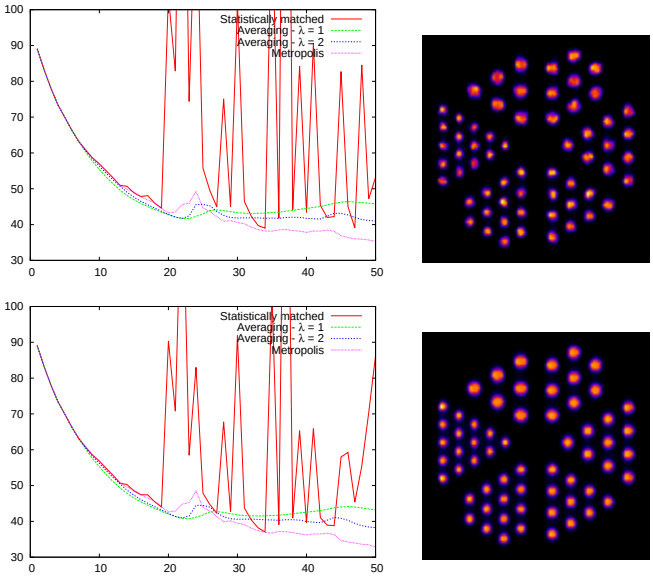


Fig. 11. Relative L_2 error curves obtained during reconstructing the 10 second (upper row) and the 1000 second (lower row) Derenzo phantoms with $N_{\text{ray}} = 1$ random ray per LOR. The cross section images are obtained with Metropolis iteration.

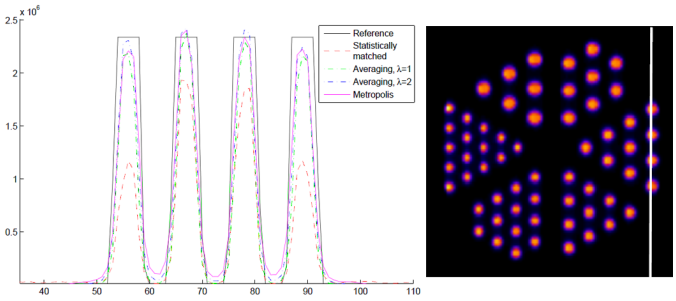


Fig. 12. Line profiles of the Derenzo 1000 sec phantom reconstructed with statistically matched, averaging and Metropolis iterations. The unit on the horizontal axis is the edge size of a voxel i.e. 0.23 mm.

We also examined the *Micro Derenzo phantom* with rod diameters $1.0, 1.1, \ldots, 1.5$ mm in different segments. The Derenzo was virtually filled with 1.6 MBq activity and we simulated a 10 sec, i.e. low-dose, and a 1000 sec, i.e. high-dose, measurement with GATE assuming ideal black detectors. A measurement is said to be low-dose when the average count per LOR is smaller than 1 since in this case the application of Poisson statistics becomes essential. The average hits per LOR are 0.05 and 5 in the 10 sec and 1000 sec measurements, respectively. If the phantom is reconstructed at $144^2 \times 128$ resolution (0.23^3 mm^3 voxels), then the size of the full system matrix is $N_{\text{LOR}} \cdot N_{\text{voxel}} \approx 10^{15}$. Ignoring scattering both in the measured object and in detectors, the SM becomes sparse but still contains 10^{10} non-zero elements. Scattering in the detectors can blur a single LOR into thousands of neighboring LORs, which increases the number of non-zero elements by orders of magnitude. Current GPU memory sizes allow to store at most 10^9 floating point values, which are already occupied by the voxel array and the LOR values, leaving not enough space for SM elements. Thus, the astronomically large number of non-zero elements renders pre-computation prohibitive and makes on-the-fly projections worth considering even for simplified projection models. Fig. 11 depicts the error and the cross section images of the Derenzo phantom reconstruction with $N_{\text{ray}} = 1$ random ray per LOR, and Fig. 12 shows a line profile of the reconstructed volume. With only $N_{\text{ray}} = 1$ line sample, the statistically matched iteration cannot correctly reconstruct the phantom, but both averaging and Metropolis iterations can since they utilize the MC estimates from more than one iteration step. Statistically matched iteration would require at least 3 line samples to have the same error curve as averaging or Metropolis iteration has with 1 line sample, i.e. averaging or Metropolis iteration offers 3 times faster projection in this case. The evaluation of the low number of samples used in a single iteration is very fast on the GPU, a full averaging or Metropolis forward projection requires 0.9 sec.

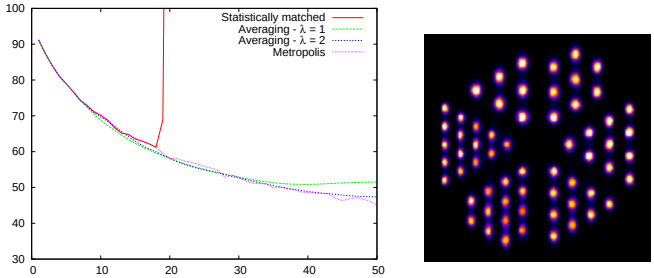


Fig. 13. Detector scattering compensation with averaging and Metropolis iterations using $N_{\text{ray}} = 1$ ray for geometric projection and $N_{\text{det}} = 64$ random LOR space offsets per LOR for MC simulation of scattering in the detector.

To test the application of averaging and Metropolis iterations in detector scattering compensation, we set LYSO crystals in the GATE simulation projecting the 1000 sec Derenzo phantom and turned on the detector model in our system as well. Detector scattering not only blurs the sinogram, but also reduces the average hits per LOR to 0.6 in the 1000 sec simulation due to the possibility that photons fly through or get lost in the detector. The results obtained with $N_{\text{det}} = 64$ 4D LOR space offsets mimicking the probability density of photon transfer in the detectors at the two ends of the LOR are shown by Fig. 13. With this number of samples, the computation time of detector blurring compensation in a single projection of the full EM iteration needs 4 sec.

B. Scattering in the measured object

Scattering in the measured object is significant in human PET, so for the purpose of examining the proposed iterations in object scatter compensation, we model the *AnyScan human PET/CT* [13]. AnyScan has 24 detector modules consisting of 27×38 crystals detectors of $3.9^2 \times 20$ mm^3. The number of LORs is $N_{\text{LOR}} = 1.5 \cdot 10^8$.

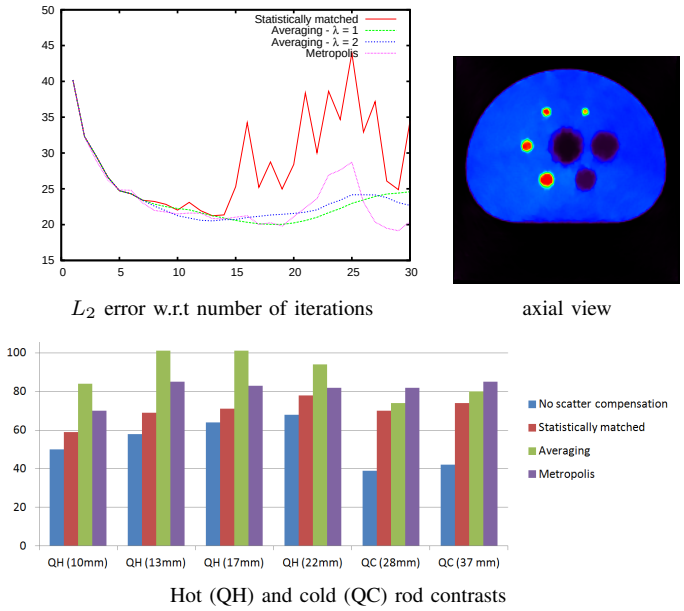


Fig. 14. Relative L_2 error curves obtained during reconstructing the NEMA human IQ phantom with $N_{\text{scatter}} = 5$ global scattering points in each iteration step, and the NEMA-NU2-2007 hot and cold contrast values after 50 iterations. The "measured data" is produced with GATE with 400-600 keV energy window. Single scatter compensation is executed in every iteration step after the 5th iteration.

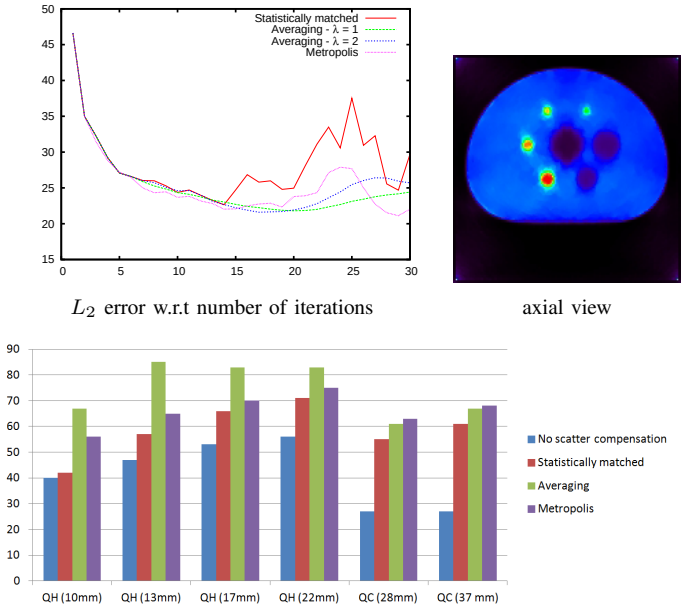


Fig. 15. Relative L_2 error curves obtained during reconstructing the NEMA human IQ phantom with $N_{\text{scatter}} = 5$ global scattering points in each iteration step. The "measured data" is produced with GATE with 100-700 keV energy window. Multiple scatter compensation is executed in every iteration step after the 5th iteration.

We used GATE to produce "measurements" of the human IQ phantom, first setting the energy discrimination window to 400–600 keV. Figure 14 shows the reconstruction at $166^2 \times 75$ voxel resolution and the NEMA-NU2-2007 contrast evaluation results with single scatter compensation taking only $N_{\text{scatter}} = 5$ random scattering point samples per iteration. The reason of selecting so few scattering point samples is to emphasize the differences of the examined iteration types. We also repeated the reconstruction for data generated by GATE with 100–700 keV energy window, approximately compensating multiple scattering as proposed in [9]. Note that all iteration types do a fairly good job in scatter compensation, but the L_2 error and contrast values are better in averaging and Metropolis iterations than in statistically matched iteration, which would require at least 10 samples to provide similar quality. Averaging iteration with $\lambda = 1$ is particularly good at improving the hot contrast. Single scatter compensation with 5 samples needs 1.1 sec in each full EM iteration step.

VI. Conclusions and Future Directions

This paper proposed the application of averaging and Metropolis iteration schemes to improve the speed and accuracy of emission tomography reconstruction working with on-the-fly MC estimates. The goal is to distribute the cost of more samples in different iteration steps, thus we get higher accuracy without increasing the computation time or storing any of the SM elements. We demonstrated the application of the method in three factored phases of a binned fully-3D PET reconstruction, including the geometric projections, scattering in the detectors during both forward and back projections, and scattering in the measured object only in forward projection. This solution has been implemented on the GPU, executing projections between hundred million voxels and LORs in a few seconds, and thus providing fully-3D reconstructions in a few minutes. The method works not only with full EM but also with OSEM and is suitable for multi-GPU implementation, which further increases the speed of the reconstruction.

We allocated the same number of samples, i.e. the same computational effort in each iteration step and distributed it evenly among voxels and LORs. However, when projections are computed on-the-fly, it is easy to allocate more samples to "difficult" voxels or LORs. It seems to be advantageous to increase the precision towards the end of the iteration, so the process will be fast at the beginning and accurate at the end. Alternatively, the precision of forward projection could be further increased by allocating more samples to those voxels that turn out to have higher activity during the reconstruction. In our future work, we examine these possibilities and try to find an optimal distribution of the total budget of samples among iterations, voxels, and LORs.

Acknowledgement

This work has been supported by the OTKA K–104476 and by TÁMOP–4.2.2.B-10/1–2010-0009.

REFERENCES

[1] K. Assié, V. Breton, I. Buvat, C. Comtat, S. Jan, M. Krieguer, D. Lazaro, Ch. Morel, M. Rey, G. Santin, L. Simon, S. Staelens, D. Strul, J.-M. Vieira, and R. V. D. Walle. Monte carlo simulation in PET and SPECT instrumentation using GATE. *Nuclear Instruments and Methods in Physics Research Section A*, 527(1–2):180–189, 2004.

[2] I. Buvat and I. Castiglioni. Monte carlo simulations in SPET and PET. *J. Nucl. Med.*, 46(1):48–61, 2002.

[3] J. Cabello and M. Rafecas. Comparison of basis functions for 3D PET reconstruction using a Monte Carlo system matrix. *Phys. Med. Biol.*, 57(7):1759–1777, 2012.

[4] R.E. Carson, W.C. Barker, J.-S. Liow, and C.A. Johnson. Design of a motion-compensation OSEM list-mode algorithm for resolution-recovery reconstruction for the HRRT. In *IEEE Nuclear Science Symposium Conference Record*, pages 3281–3285, 2003.

[5] S. Jan et al. GATE: A simulation toolkit for PET and SPECT. *Phys. Med. Biol.*, 49(19):4543–4561, 2004.

[6] P. Joseph. An improved algorithm for reprojecting rays through pixel images. *IEEE Trans. Med. Imaging*, 26(1):192–196, 2007.

[7] K. S. Kim and J. C. Ye. Fully 3D iterative scatter-corrected OSEM for HRRT PET using GPU. *Phys. Med. Biol.*, 56:4991–5009, 2011.

[8] J. Lantos, Sz. Czifrus, D. Légrády, and A. Cserkaszky. Detector response function of the NanoPET/CT system. In *IEEE Nuclear Science Symposium and Medical Imaging Conference*, 2010.

[9] M. Magdics, L. Szirmay-Kalos, B. Tóth, and T. Bükki. Higher order scattering estimation for PET. In *Medical Imaging Conference*, MIC, 2012.

[10] M. Magdics, L. Szirmay-Kalos, B. Tóth, and T. Umenhoffer. Filtered Sampling for PET. In *Medical Imaging Conference*, MIC, 2012.

[11] M. Magdics, L. Szirmay-Kalos, B. Tóth, Á. Csendesi, and A. Penzov. Scatter estimation for PET reconstruction. In *Proceedings of the 7th International Conference on Numerical Methods and Applications*, NMA'10, pages 77–86, 2011.

[12] M. Magdics et al. Detector modeling techniques for pre-clinical 3D PET reconstruction on the GPU. In *Full 3D Concerence*, pages 375–8, 2011.

[13] Mediso. AnyScan PET/CT. http://www.mediso.com/products.php?type=hw &fid=1&pid=73.

[14] Mediso. NanoScan-PET/CT. http://www.mediso.com/products.php?type=hw &fid=2&pid=86.

[15] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1091, 1953.

[16] S. Moehrs, M. Defrise, N. Belcari, A. D. Guerra, A. Bartoli, S. Fabbri, and G. Zanetti. Multi-ray-based system matrix generation for 3D PET reconstruction. *Phys. Med. Biol.*, 53:6925–6945, 2008.

[17] V.Y. Panin, F. Kehren, C. Michel, and M. Casey. Fully 3-D PET reconstruction with system matrix derived from point source measurements. *IEEE Trans. Med. Imaging*, 25(7):907–921, july 2006.

[18] G. Pratx and C. Levin. Online detector response calculations for high-resolution PET image reconstruction. *Phys. Med. Biol.*, 56(13):4023–4040, 2011.

[19] J. Qi, R. M. Leahy, S. R. Cherry, A. Chatziioannou, and T. H. Farquhar. High-resolution 3D Bayesian image reconstruction using the microPET small-animal scanner. *Phys. Med. Biol.*, 43(4):1001, 1998.

[20] M. Rafecas, B. Mosler, M. Dietz, M. Pögl, A. Stamatakis, D. P. McElroy, and S. I. Ziegler. Use of a Monte Carlo-based probability matrix for 3-D iterative reconstruction of MADPET-II data. *IEEE Trans. on Nucl. Sci.*, 51(5):2597, 2004.

[21] A. J. Reader and H. Zaidi. Advances in PET image reconstruction. *PET Clinics*, 2(2):173 – 190, 2007.

[22] L. Shepp and Y. Vardi. Maximum likelihood reconstruction for emission tomography. *IEEE Trans. Med. Imaging*, 1:113–122, 1982.

[23] R. L. Siddon. Fast calculation of the exact radiological path for a three-dimensional ct array. *Medical Physics*, 12(2):252–257, 1985.

[24] L. Szirmay-Kalos, B. Tóth, and M. Magdics. Free Path Sampling in High Resolution Inhomogeneous Participating Media. *Computer Graphics Forum*, 30(1):85–97, 2011.

[25] C.C. Watson, D. Newport, and M.E. Casey. *Three-Dimensional Image Reconstruction in Radiation and Nuclear Medicine*, chapter A single scattering simulation technique for scatter correction in 3D PET, pages 255–268. Kluwer Academic Publishers, 1996.

[26] A. Wirth, A. Cserkaszky, B. Kári, D. Legrády, S. Fehér, S. Czifrus, and B. Domonkos. Implementation of 3D Monte Carlo PET reconstruction algorithm on GPU. In *IEEE Nuclear Science Symposium Conference Record*, NSS/MIC, pages 4106–4109, 2009.