Physics in Medicine and Biology

Learning SPECT detector angular response function with neural network for accelerating Monte-Carlo simulations

To cite this article before publication: David Sarrut *et al* 2018 *Phys. Med. Biol.* in press https://doi.org/10.1088/1361-6560/aae331

Learning SPECT detector angular response function with neural network for accelerating Monte-Carlo simulations

D. Sarrut[1], N. Krah[1], JN. Badel[1], JM. Létang[1]

[1] Université de Lyon, CREATIS; CNRS UMR5220; Inserm U1044; INSA-Lyon; Université Lyon 1; Centre Léon Bérard, France.

E-mail: `david.sarrut@creatis.insa-lyon.fr`

Abstract.

A method to speed-up Monte-Carlo simulations of Single Photon Emission Computed Tomography (SPECT) imaging is proposed. It uses an artificial neural network (ANN) to learn the Angular Response Function (ARF) of a collimator-detector system. The ANN is trained once from a complete simulation including the complete detector head with collimator, crystal, and digitization process. During the simulation, particle tracking inside the SPECT head is replaced by a plane. Photons are stopped at the plane and the energy and direction are used as input to the ANN that provide detection probabilities in each energy windows. Compared to histogram-based ARF, the proposed method is less dependent on the statistics of the training data, provides similar simulation efficiency, and requires less training data. The implementation is available within the GATE platform.

1. Introduction

SPECT imaging is widely used to provide 3D images of the spatial distribution of single-photon emission radiotracers. A conventional SPECT system is composed of a scintillation detector and photomultiplier tubes (PMTs) to record the location and the energy of detected photons. In front of the detector, a lead or tungsten honeycomb collimator is used to select photons traveling along a given direction in order to retrieve an estimate of their point of origin within the patient. The detected photons are stored according to energy windows defined either around the radionuclide photo-peaks and such to account for lower-energy photons that have most likely undergone Compton scatter prior to detection (and whose origin is therefore uncertain).

Monte-Carlo simulation of SPECT images is typically done in two main steps: 1) tracking the particles inside the medium, e.g., a patient CT image, and 2) tracking the particles in the SPECT detector head. The first step generates particles from an activity distribution of a given radionuclide such as ^{99m}Tc, ^{111}In or ^{177}Lu and tracks photons from voxel to voxel until they escape from the patient. This process may be accelerated by variance reduction techniques (VRT) such as ray-tracing based methods [1, 2, 3] and/or the use of GPU [4, 5]. Here, we focus on the second step that involves the simulation of the photon interactions in the collimator and in the scintillator (crystal) as well as the digitization chain of the readout electronic components. The collimator-detector response function (CDRF) combines the accumulated effects of all interactions in the imaging head. It may be approximated with ARF methods [6, 7, 5].

The ARF method replaces the explicit photon tracking in the imaging head by a tabulated model of the CDRF. The tabulated model is derived from a simulation with a source of gamma covering the energy range of the radionuclide of interest and including the complete detector head with collimator, crystal and digitization process. The model takes as input the direction angles and the energy of an incoming photon and determines the probability of this photon to be detected in each defined energy window. This first step needs to be performed only once per type of SPECT head, radionuclide and defined energy windows. Once the lookup tables are computed, they can be used for every simulation having the same conditions (same collimator/detector, radionuclide energy windows), independently of the source distribution and the medium, phantom or patient. The ARF method assumes that a photon which interacts with the collimator will be detected at its geometric intersection point on the detector plane, taking into account the spatial uncertainty. This approximation has been shown to be sufficient [6] and will not be further studied. Furthermore, detector dead-time is neglected in this work.

The ARF approach has been shown to be efficient and to provide variance reduction that speeds up the simulation [7]. The exact speedup factor depends on several factors (radionuclide, energy windows etc.), but has been estimated between 20 and 100 [7]. Those speedup factors, however, have been evaluated by taking into account also the time spent to track of the photons inside the CT image. To our knowledge, the intrinsic efficiency of the ARF method, which depends on the number of counts and the energy, has not explicitly been evaluated.

One limitation of the ARF method is that computing the tabulated functions is a lengthy process: about 10^9 to 10^{11} emitted primary photons are required to obtain a good statistical uncertainty for a given energy window [7, 5]. In the Gate implementation proposed in [7], a table is required for every digitizer window used in the simulation, and all energy windows are discretized into several energy bins. In [5], the tables are computed with more than 10^{11} photons, using a voxelized model of the collimator and without including the scattering in the crystal. In practice, the process to compute ARF tables remains cumbersome because it implies numerous long simulations to be performed and gathered.

In this paper, to speed Monte-Carlo simulations, we propose to use ARF computed by a Neural Network (NN) trained with data obtained from simulations, in combination with russian-roulette variance reduction technique.

2. Method

We investigated the feasibility of using a Neural Network (NN) trained with data obtained from simulations to replace ARF tables. We will refer to the ARF using NN instead of tables as ARF-nn, while the original tabulated ARF model will be called ARF-histo. Once trained, the ARF-nn is used like the ARF-histo in order to provide the probability that a given photon with given incident angles and energy are counted in an energy window. The first step is to train the NN, and the second step is to use it to compute an image. All developments were performed in the open-source Gate platform [8] based on Geant4 [9] and will be available in the next release.

2.1. Building the training dataset

The training dataset was generated from a Monte-Carlo simulation. The geometry was composed of a complete SPECT head description with collimator, crystal and digitizer chain.

Source The photon source was defined as a plane in front of the collimator which covered the whole detection head. The photon energy was drawn from an uniform distribution with maximum energy the largest energy of the considered radionuclide. The directions of the emitted particle were sampled from an isotropic uniform distribution. Low energy photons with large angle directions are almost always absorbed by the collimator. Therefore, the angular distribution may be limited to a range for which the detection probability was not zero. For high energy photons which have a non negligible probability to penetrate septa, such as ^{131}I, the full angular distribution was taken into account.

Detection To build the training dataset, the simulation records four values for each photon reaching the detector: the incident angles θ and ϕ, the incident energy E, and the label of the energy window in which the photon was detected. Angles are expressed in spherical coordinates, in accordance with the convention in [7, 6]. An additional energy window corresponding to a "non-detected" photon was included. For example, we used 8 windows for ^{131}I, two corresponding to the photo-peaks denoted W_{Peak}, five for scatter windows around and in-between the photo-peaks denoted W_{SC}, and one "non-detected" window denoted W_{ND}.

Russian-roulette Most of the photons are absorbed in the collimator and end up in the W_{ND} window. For example, we computed that less than 10^{-4} of photons emitted isotropically at 364 keV from a ^{131}I source will be detected in the W_{Peak} window. Hence, at the end of the simulation the number of counts (detected photons) would be very unbalanced between the different energy windows. Russian-Roulette (RR) variance reduction technique was used to mitigate this effect. Specifically, only 1 out of w counts in the W_{ND} window was stored with a weight of w. This method allowed us to reduce the size of the output dataset without biasing the probabilities.

2.2. Training the Neural Network

Machine learning with Artificial Neural Networks (ANN) recently (re)gained attention in a large variety of tasks, notably natural image classification. An ANN works as a mapping function from an input space to an output space. Inspired from biological neural networks, the mapping function is in the form of a network of weighted additive values with non-linear (but differentiable) transfer functions, organized in layers of neurons. Training the network consists in optimizing a loss function that minimizes some sort of difference between the known and predicted output, performed on a large training dataset.

Here, the 3D input space $\mathcal{X}$ is spanned by the two angles θ and ϕ, and the energy E. The input fed to the NN is an $N \times 3$ matrix X containing N vectors $\mathbf{x} = (\theta, \phi, E) \in \mathcal{X}$, one for each photon. The goal of the NN is to predict the output vector $\mathbf{y}$, being the probability of the input photon to be detected in one energy window. In the so-called "one-hot" notation, $\mathbf{y}$ is a vector of size n (the number of energy window), and the output space is denoted by $\mathcal{Y}$. The vector component y_i indicates the probability of the photon to be detected in the i^{th} upright energy window. The purpose of the NN is to find a mapping h from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$. The training dataset consists of (X, Y), with Y the collection of $\mathbf{y}$. Training the NN consists in finding the neuron weights for which h optimally represents the training dataset.

The NN architecture was the following. We used $H = 400$ neurons in each of 3 hidden linear fully connected layers. The activation function was the Rectified Linear Unit (ReLu) $r(x) = max(0, x)$. Each layer l has a matrix of weights $\mathbf{w}_l$ and computes $r(\mathbf{w}_l \mathbf{x})$. Combined together, the network becomes $h(\mathbf{x}) = r(\mathbf{w}_3(r(\mathbf{w}_2(r(\mathbf{w}_1\mathbf{x}))))) = \mathbf{y}$. Because the layers are fully connected, the NN has $3\,H + H^2 + H\,N_W$ weight values, with 3 input data dimensions (E, θ, ϕ) and N_W the number of energy windows. The optimisation process aims to find the values of the weights $\mathbf{w}$ that minimize a difference (loss) between the input dataset $\mathbf{y}_{data}$ and the output of the network $h(\mathbf{x}) = \mathbf{y}$. The loss function was the multiclass *cross entropy* between the two probability distributions $p_{y_{data}}$ and p_y. This criterion encourages the model to assign higher probability values to the correct labels across the training samples. It combines the negative log-likelihood loss with the normalized exponential activation function (the *softmax* function), that normalizes the exponential probabilities between 0 and 1.

The optimization was performed using Adam optimizer which is a first-order gradient-based optimization based on adaptive estimates of lower-order moments [10]. At each iteration (referred to as *epoch*), back-propagation is used to calculate the derivative of the loss function with respect to each weight and a fraction of this derivative is subtracted from that weight. The fraction is determined by the learning rate α, which controls the balance between convergence speed and precision. We used an adaptive learning rate starting at $\alpha = 0.0001$ and decaying by a constant factor when improvement in the cost function is lower than a given value. A maximum of 1000 iterations with stochastic batches of 5000 samples at each iteration was used. Input data X where normalized according to mean and standard deviation. At the end of the optimisation, we ended up with a set of about 1.6×10^5 weights which define the NN (e.g. with $N_W = 8$).

2.3. Generating an image with ARF-nn

The ARF-nn method was decomposed into a two-step process. First, as with ARF-histo, the simulation is run with the SPECT head replaced by an empty plane of 1 nm thickness, henceforth called the "ARF plane", located just in front of the collimator. The position, direction, and energy of the photons that reach the ARF plane are stored in a dataset. In a second step, performed after the simulation, the image is computed as follows using this dataset. For every photon, the coordinates (u, v) in the image plane, sampled with 4×4 mm pixel size, is determined from the position in the ARF plane as proposed in [6], *i.e.* using the point where the incident photon direction vector intersects the image plane located half-way of the crystal length. Figure 1 illustrates the process with the ARF and image planes. For the values (E, θ, ϕ) of each incident photon, the NN is used to get the probabilities $h(E, \theta, \phi) = y_i$. $I(u, v, i)$ is then incremented by y_i, with i the index of the energy window.

The time gain of the method compared to Monte-Carlo is that 1) the simulation required to generate the image is expected to be faster than a full simulation including tracking in the detector head, and 2) the ARF model (histograms or NN) gives the probability in all energy windows thus contributing to variance reduction.

2.4. Implementation

The simulation was implemented in Gate version 8.0 [8], using Geant4 version 10.3. A new actor was provided, called `Gate_NN_ARF_Actor`. When the flag `train_mode` is set, it stores a training dataset composed of the input parameters θ, ϕ, E and the corresponding energy window output i. The coordinates of the photons are not stored in this phase. When the flag `test_mode` is set, the actor stores the parameters u, v, θ, ϕ, E that will be used by the ARF model (with u and v the coordinates of photons in the ARF plane). All output files are in raw binary file format. Neural network operations (training, image generation) were implemented in Python with the pytorch framework [11] using CUDA GPU acceleration. Source code is open-source and will be available in the next Gate release. All computations were performed on an Intel Xeon CPU E5-2640 v4 @ 2.40GHz, and an NVIDIA Titan Xp (GP102-450-A1) with 12 GB memory.

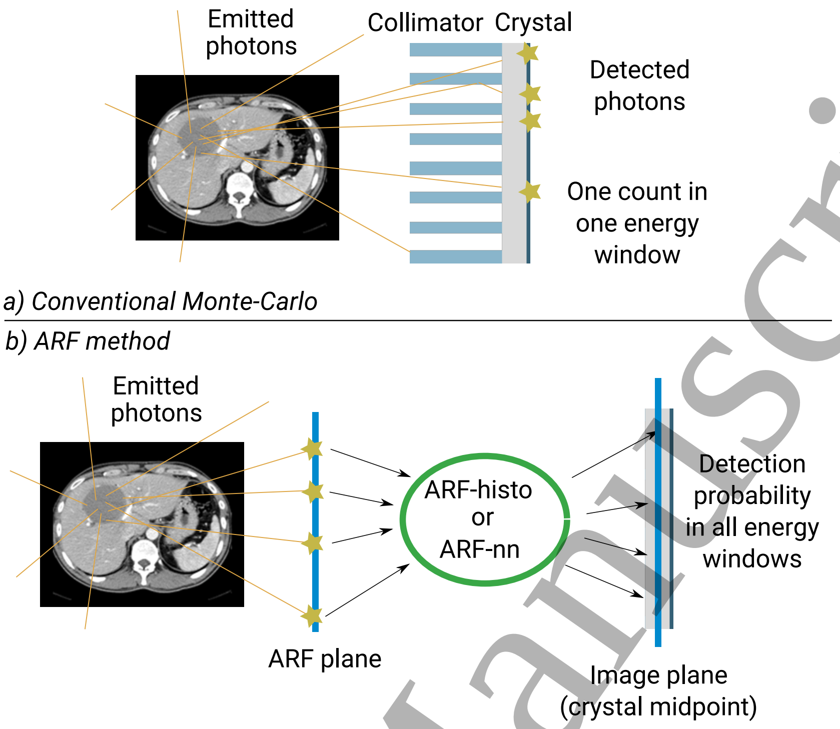
Emitted photons
Collimator
Crystal
Detected photons
One count in one energy window
a) Conventional Monte-Carlo
b) ARF method
Emitted photons
ARF-histo or ARF-nn
Detection probability in all energy windows
ARF plane
Image plane (crystal midpoint)

Figure 1. Top: Conventional SPECT simulation principle with photons tracked in collimator and crystal. Bottom: SPECT simulation with ARF method (both ARF-histo or ARF-nn). Photons are stopped at the ARF plane and the ARF model provides probabilities to detect the photons in each energy window (distance between ARF plane and image plane not at scale).

3. Simulation tests and validation

We evaluated the accuracy and efficiency of both ARF methods compared to brute-force Monte-Carlo method without variance reduction technique which we called *analog* Monte-Carlo. We further studied the influence of the training dataset size and histogram binning. The following paragraphs describe the two types of simulations, one to the generate the training dataset and one to generate the data from which the image is built.

Training datasets We simulated a head of the GE Discovery 670 with NaI(Tl) crystal equipped with Medium Energy General Purpose (MEGP) and High Energy General Purpose (HEGP) parallel-hole collimators, respectively, notably used for ^{111}In and ^{131}I sources. Hole diameters were 3 and 4 mm, respectively, with a septal thickness of 1.05 and 1.8 mm. Crystal thickness was 9.525 mm (3/8 inch). The effect of the digitizer chain was modeled by applying a spatial Gaussian blurring of 3.97 mm [12]. Table 1 displays the energy windows considered, 5 windows for ^{111}In and 7 for ^{131}I. The source was defined as described in section 2.1 with photon energy ranges [150 − 270] and [200 − 780] keV, respectively. The source plane was positioned at the collimator entrance. The russian-roulette factor was set to $w = 40$, i.e., only 1/40 of the photons which did not traverse the collimator were stored. One billion primary photons were generated. Two NN were trained, one for each collimator/digitizer configuration, with the same set of parameters.

Radionuclide	Windows	Energy ranges
^{111}In	W_{Peak} 171 keV	[156 ; 186] keV
(MEGP)	W_{Peak} 245 keV	[224 ; 272] keV
	W_{SC} 1,2,3	[150 ; 156] [186 ; 192] [218 ; 224] keV
^{131}I	W_{Peak} 364.5 keV	[336 ; 392] keV
(HEGP)	W_{Peak} 637 keV	[595 ; 679] keV
	W_{Peak} 722 keV	[679 ; 765] keV
	W_{SC} 1,2,3,4	[314 ; 336] [392 ; 414] [414 ; 556] [556 ; 595] keV

Table 1. Energy windows for ^{111}In (MEGP) and ^{131}I(HEGP) simulations.

Test datasets Three tests were performed, each with a different source. In Test1, one isotropic spherical source of 1 cm radius and 364 keV energy was used. In Test2, the source was composed of spheres of 4 cm radius with energies corresponding to the ^{131}I energy windows described previously. The sources were linearly positioned at 20 cm distance from the SPECT head, regularly spaced apart by 7 cm. The source geometry in Test3 was identical to Test2, but with energies corresponding to the ^{111}In energy windows. Test1 and Test2 were performed with HEGP, and Test3 with MEGP collimators. All tests were done in air without attenuating media.

For each test, two types of simulations were performed: one reference analog Monte-Carlo simulation which included the geometry of the detector, in particular the collimator; and one simulation in which the detector head was replaced by a simple detector plane. Reference analog Monte-Carlo simulations were performed with 2×10^9 to 2×10^{10} primary photons. The output data were projection images containing the same number of channels as the number of energy windows. For both ARF methods, i.e., NN and histogram, the simulation was performed to record photon positions, angles, and energy in the ARF detector plane. ARF-histo generated images were built using different numbers of bins. We selected the following numbers of bins: 200, 150, 120, 100, 80, 50, 25, 12, used for all three parameters θ, ϕ and E to study the effect of the bins size on the constructed image.

End to end simulation under realistic conditions For illustration purpose, we also performed Test4, a simulation of a patient CT image with a voxelised ^{131}I source having uniform activity located in the patient tyroid. The image and the source were sampled at $3 \times 3 \times 3$ mm^3. The imaging system was the GE Discovery 670 with HEGP collimator. 10^{10} primary photons corresponding to the ^{131}I photo peaks were simulated, 6.227% at 284.31 keV, 81.3% at 364.49 keV, and 7.132% at 636.99 keV. Tracking cuts were set to 1 m inside the patient to avoid generating and tracking electrons in this region, and were set to 1 mm in the SPECT head.

Evaluation criterion We evaluated the difference between images generated with analog Monte-Carlo and ARF methods as the Mean Absolute Error δ (MAE) for all pixels having more than 5% of the maximum counts in the image, denoted $P_{5\%}$. In equation 1 below, c_k and c_k^{ref} refer to the number of counts in pixel k computed with ARF or reference analog Monte-Carlo, and $|P_{5\%}|$ is the cardinality of the set $P_{5\%}$. The evolution of this error as a function of training dataset size and binning parameters were also studied.

$$\text{(MAE)} \qquad \delta = \frac{1}{|P_{5\%}|} \sum_{k}^{P_{5\%}} |c_k^{ref} - c_k| \tag{1}$$

We computed the efficiency of the ARF variance reduction technique. Photon detection in SPECT is a Poisson process and the statistical uncertainty of the analog Monte-Carlo simulation is the square root of the number of counts. The relative uncertainty can therefore be estimated at the end of a simulation for all pixels in the images as: $\sigma(k) = \sqrt{c_k}/c_k$. The global uncertainty is defined as the mean uncertainty over all pixels in $P_{5\%}$. As a control, it was checked that this method led to similar uncertainty estimation as with the conventional batch method. However, for the ARF methods, detected photons are accumulated with scalar weights not integer counts and the uncertainty cannot be estimated after the simulation. We calculated the uncertainty σ_k according to equation 2 below following the history by history method [13], where N is the number of primary events in the simulation, i a given event, and $c_{k,i}$ the count probability, given by the ARF, of event i in pixel k. The global uncertainty σ is averaged over all pixels in $P_{5\%}$. Here also, we verified that this history by history method led to a similar uncertainty estimation as with the conventional batch method. To our knowledge, the efficiency of the ARF method was never explicitly quantified. Like in [3], the efficiency ε_k of a method in pixel k is computed with equation 4 taking into account the computed time t and the uncertainty, and the mean efficiency ε_{mean} for the entire image is calculated by averaging over all voxels in $P_{5\%}$. The speedup of one method compared to another was computed as the pixelwise ratio of efficiencies ε_k.

Note that for Test1, Test2, and Test3, the efficiency considers the computation time for particles tracked inside the detector head, without taking into account the time required to track particles in the phantom or in the patient.

$$\sigma_k = \sqrt{\frac{1}{N-1}\left(\frac{\sum_i^N c_{k,i}^2}{N} - \left(\frac{\sum_i^N c_{k,i}}{N}\right)^2\right)} \tag{2}$$

$$\sigma_{global} = \frac{1}{|P_{5\%}|} \sum_{k \in P_{5\%}} \sigma_k \tag{3}$$

$$\varepsilon_k = \frac{1}{t \times \sigma_k^2} \quad ; \quad \varepsilon_{mean} = \frac{1}{|P_{5\%}|} \sum_{k \in P_{5\%}} \varepsilon_k \tag{4}$$

Noise analysis The analog Monte-Carlo method accumulates discrete counts. Hence, the variation of the counts in a pixel follows a Poisson distribution, with the mean counts equal to the variance. However, the ARF method is a variance reduction technique that accumulates fractions of counts. Hence, the distribution of counts is not expected to be Poissonian. In order to analyze the difference between the noise distributions of analog and ARF methods, simulations of a 20 cm radius circular source with uniform energy distribution in the range $[336 - 392]$ keV were performed. Such simulations generate images with a large area containing homogeneous values, where the count distribution can be plotted and analyzed. Count values were sampled from pixels in the area and binned to form a histogram.

4. Results

Figures 2, 3, and 4 depict MAE errors for the three tests for different histogram binnings and sizes of the training datasets. Figure 5 shows the images obtained in Test1 for the W_{Peak} 364 keV window , with a saturated colorscale illustrating the typical ”star” effect of ^{131}I and HEGP collimator.

Table 2 depicts the mean speedup compared to the analog Monte-Carlo, averaged over all energy windows and all pixels containing more than 5% of the maximum counts, for the three test cases and the three simulation methods. The number of primaries and the mean uncertainty is also reported. Examples of simulated images are shown in figure 6 which also depicts an image of speedup per pixel. The associated statistical uncertainty per pixel is depicted figure 7.

In the training dataset computed with 10^9 particles, only about 25×10^6 particles were stored thanks to the Russian-Roulette. Approximately 99.4% of the photons did not lead to a detected event. The training dataset stored file was 783 MB. The NN training phase lead to files smaller than 2 MB. The histogram files were between 100 kB (binning equal to 12) to 305 MB (binning equal to 200).

Regarding noise analysis, figure 8 displays the count distribution in a homogeneous image area obtained from the simulation of a uniform energy source. The analog Monte-Carlo simulation reproduced the Poisson noise properties, with distribution’s mean value equal to its variance for the three tested numbers of primary particles: 5×10^8, 10^9 and 2×10^9. Fitted Poisson are also shown in the figure. For the ARF method, the mean counts were very close to the analog mean counts (less than 1% difference), but the distributions were not Poisson. They may be approximated by a Gaussian distribution (black dashed line in the figure). In this case, ARF distributions were obtained for 5×10^7 particles and scaled to the number of particles of the analog simulation. If realistic noise is needed, the ARF image should be scaled to the expected number of primary particles and pixel values perturbed by stochastic samples drawn from a Poisson distribution with mean the current pixel counts.

Figure 9 illustrates the complete simulation with patient CT and ^{131}I source. Image profiles of the analog Monte-Carlo and ARF-nn simulations were compared. Note that the pixel values were not scaled and absolute count values were depicted. The effect of Poisson noise is illustrated in the second row.

Table 3 summarizes the simulation times with and without tracking in the detector. The second phase of ARF image generation that uses ARF tables to create the image took about 100 seconds for ARF-nn and 200 seconds for ARF-histo for 10^8 primaries. However, this latter comparison was not fair as the vectorial operations of neural network were performed on GPU, while the histogram-based task was on CPU. The training of the neural network took between 35-45 minutes according to the dataset sizes. For Test4 with the patient CT, most of the time is spent in tracking the particles inside the CT image and the speed decreases to about 5,000 particles per seconds (PPS). Statistical uncertainties were comparable between analog with 10^{10} primary particles and ARF-nn with 4×10^7 particles, leading to a 90 to 400 times higher efficiency depending on the energy window.

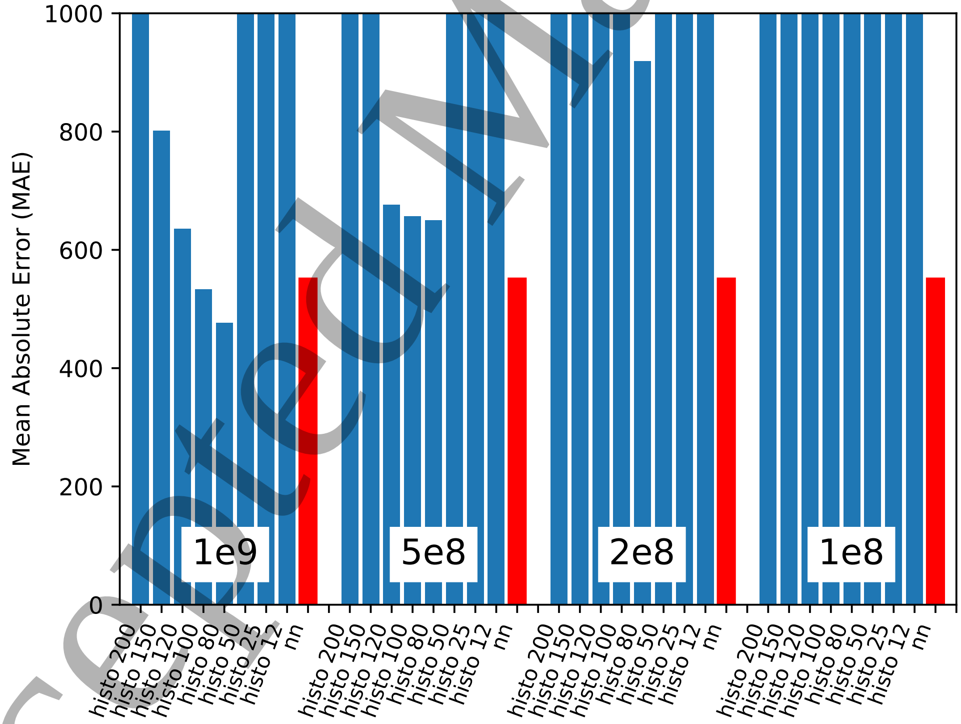

Figure 2. Mean Absolue Errors (MAE see eq. 1) results for Test1 (single peak source, HEGP) for ARF methods with several histogram binnings (in blue), with ARF-nn (in red), and for four training dataset sizes ($10^9, 5 \times 10^8, 2 \times 10^8$ and 10^8).

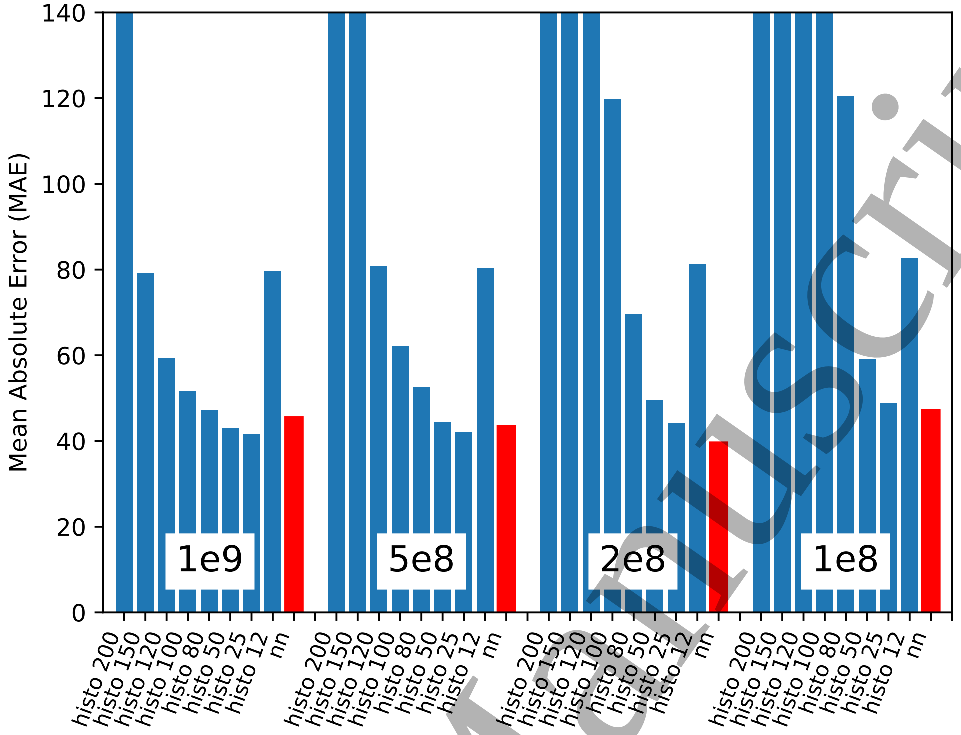

Figure 3. Mean Absolute Errors (MAE see eq. 1) results for Test2 (HEGP, ^{131}I) for ARF methods with several histogram binnings (in blue), with ARF-nn (in red), and for four training dataset sizes ($10^9, 5 \times 10^8, 2 \times 10^8$ and 10^8).

Method	N	Uncertainty	Speedup
Test1 reference	2e10	<3%	1
Test1 ARF-histo	1e7	8-10%	55-198
Test1 ARF-nn	1e7	8-10%	62-264
Test2 reference	4e10	4-6%	1
Test2 ARF-histo	4e7	4-6%	1400-3100 (low) / 100-300 (high)
Test2 ARF-nn	4e7	4-6%	1400-3400 (low) / 100-300 (high)
Test3 reference	2e9	9-13%	1
Test3 ARF-histo	1e8	8-10%	17-50
Test3 ARF-nn	1e8	8-10%	18-49
Test4 reference	1e10	7-10%	1
Test4 ARF-histo	4e7	5-9%	90-400
Test4 ARF-nn	4e7	5-9%	89-390

Table 2. Range of obtained speedups in the energy windows between analog and ARF methods for the three test cases. The number of particules used and the mean statistical uncertainty (eq. 2) is also depicted. For Test2, we indicate speedup (ratio of ε, eq. 4) in high count regions (more than 1000) and low count regions (around 200).

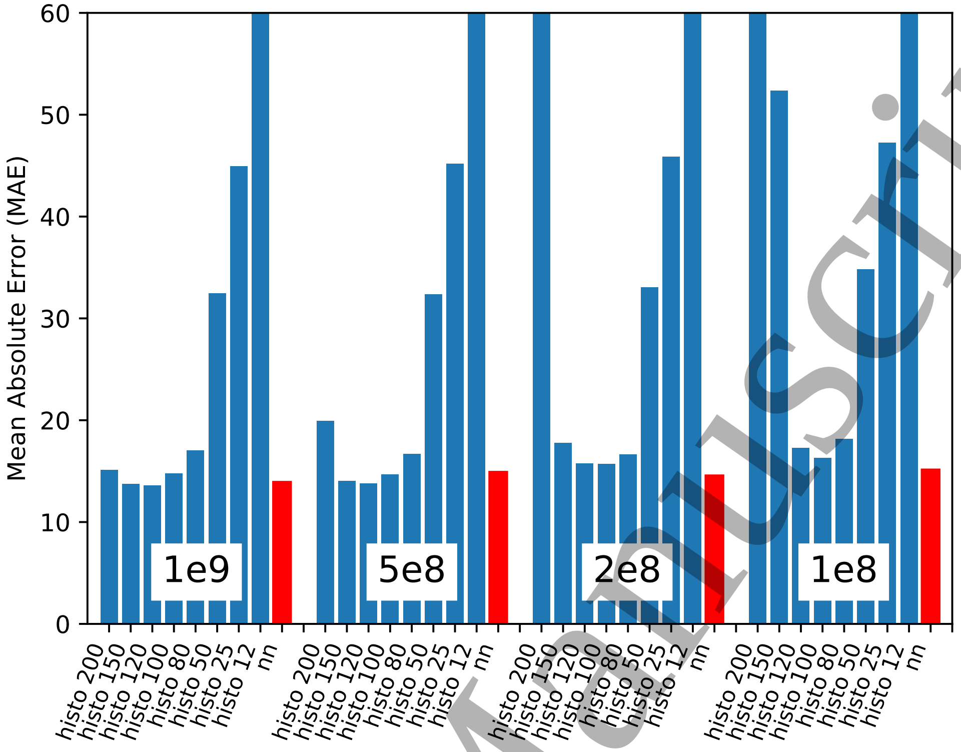

Figure 4. Mean Absolue Errors (MAE see eq. 1) results for Test3 (MEGP, ^{111}In) for ARF methods with several histogram binnings (in blue), with ARF-nn (in red), and for four training dataset sizes ($10^9, 5 \times 10^8, 2 \times 10^8$ and 10^8).

Simulation	Reference	ARF
Test1 (HEGP) PPS	76,000	90,000
Test2 (HEGP) PPS	50,000	90,000
Test3 (MEGP) PPS	36,000	46,000
Test4 (Patient) PPS	5,120	5,430

Table 3. Computation time in PPS (Particle Per Seconds) for the different types of simulation. The column "reference" depicts PPS for the complete Monte-Carlo simulation including tracking in the detector. The column "ARF" depicts PPS for ARF simulations (no collimator, no crystal).

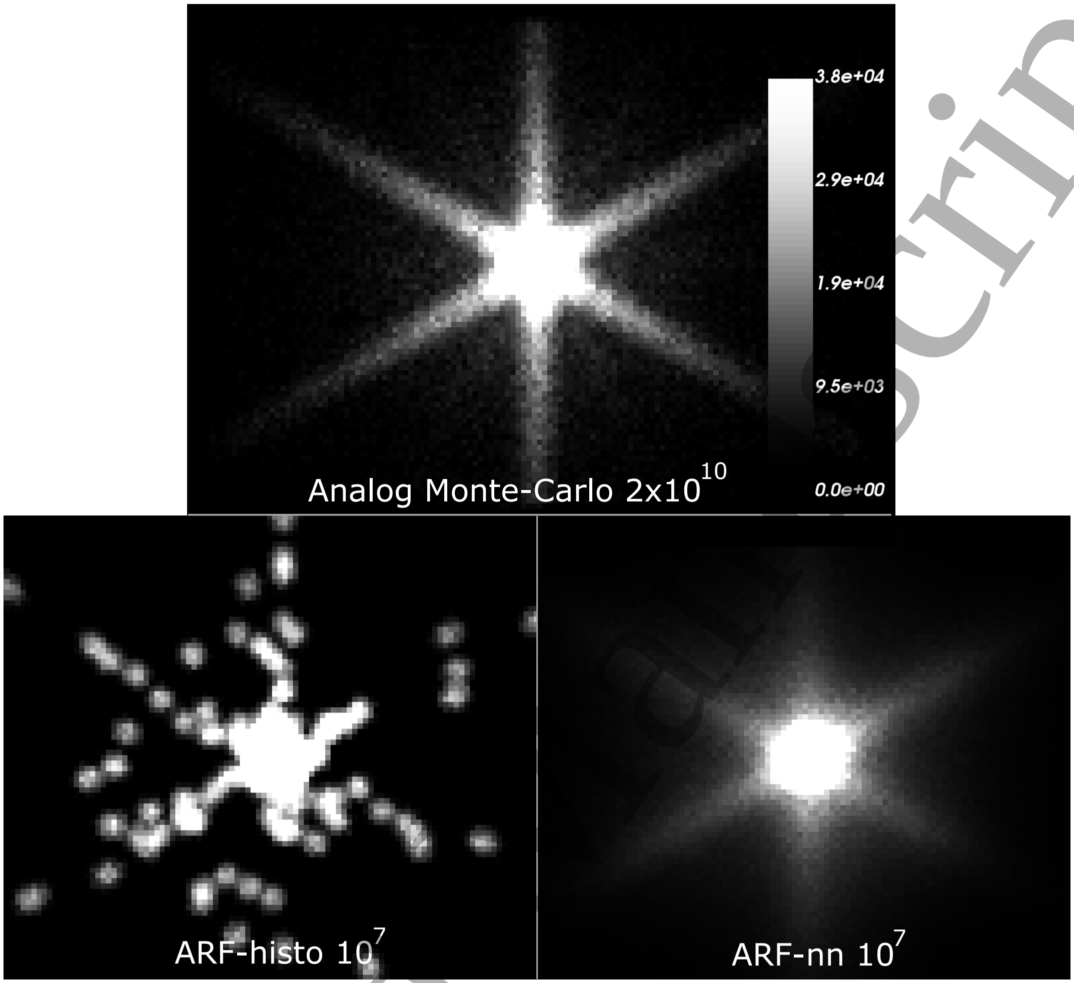

Figure 5. Typical star effect of the HEGP collimator for high energy gamma, clearly visible with the analog Monte-Carlo simulation (2×10^{10} particles, top image), visible with ARF-nn (10^7 particles, bottom right) and not visible with ARF-histo (10^7 particles, bottom left). Colorscale is saturated and is the same for all three images.

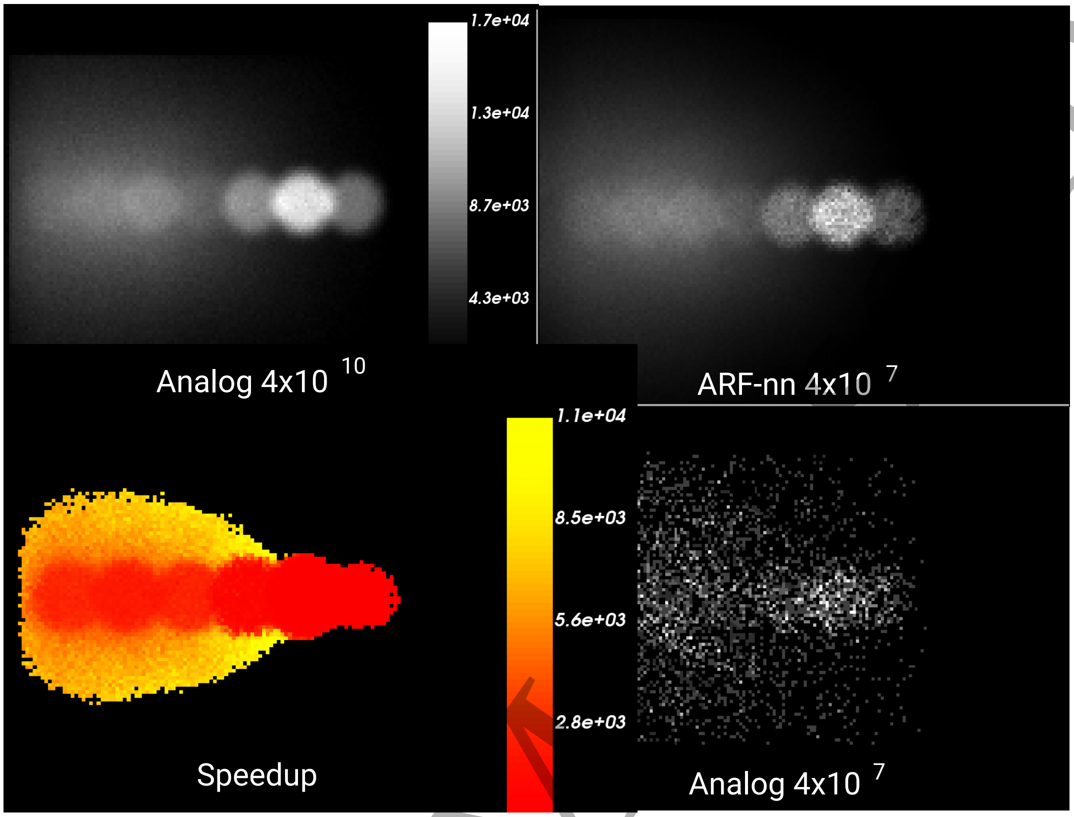

Figure 6. Top-left: simulated SPECT image for test2 (HEGP, ^{131}I) for W_{SC} [392-414] keV window. The simulation was obtained with analog Monte-Carlo, 4×10^{10} primaries. Several sources are visible, the maximum number of counts in a voxel was 17392. Top-right: simulated SPECT image for the same simulation, but performed with ARF-nn and only 4×10^{7} primaries. By comparison, analog Monte-Carlo with 4×10^{7} is shown bottom-right. Bottom left: speedup per voxel between analog and ARF-nn, for the same image. High count areas depicted speedup around 200, while low count regions may reach a speedup around 10^4.

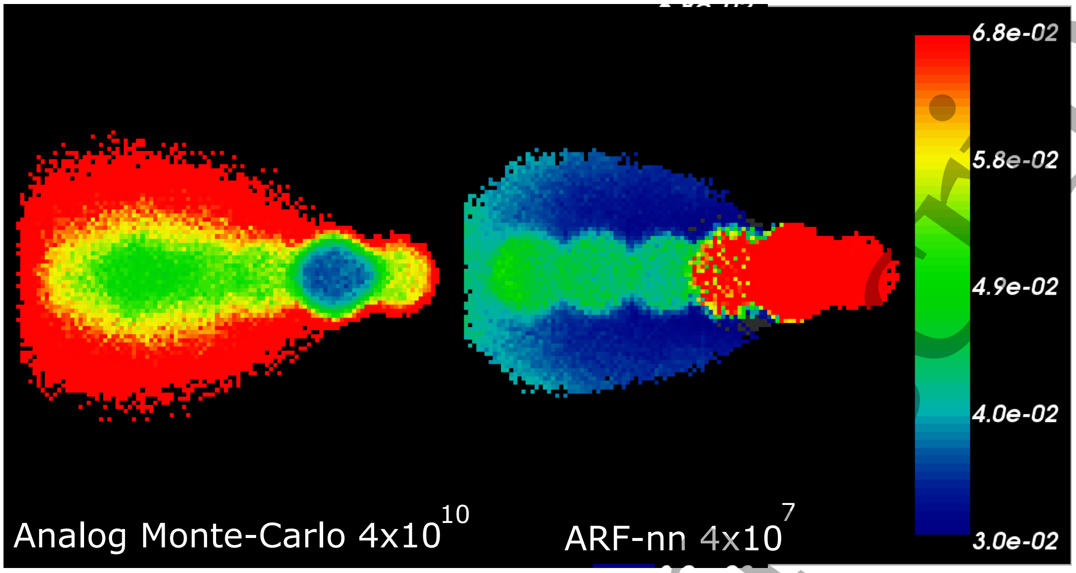

Figure 7. Statistical uncertainty was on average similar for analog (4×10^{10} primaries) and ARF-nn (4×10^7 primaries), but differently distributed in the image. Left: statistical uncertainty for analog Monte-Carlo. Right: for ARF-nn. With analog Monte-Carlo, high count regions were associated with low uncertainty and convergence was much slower in low count regions. For ARF-nn, convergence is much faster in low count region.

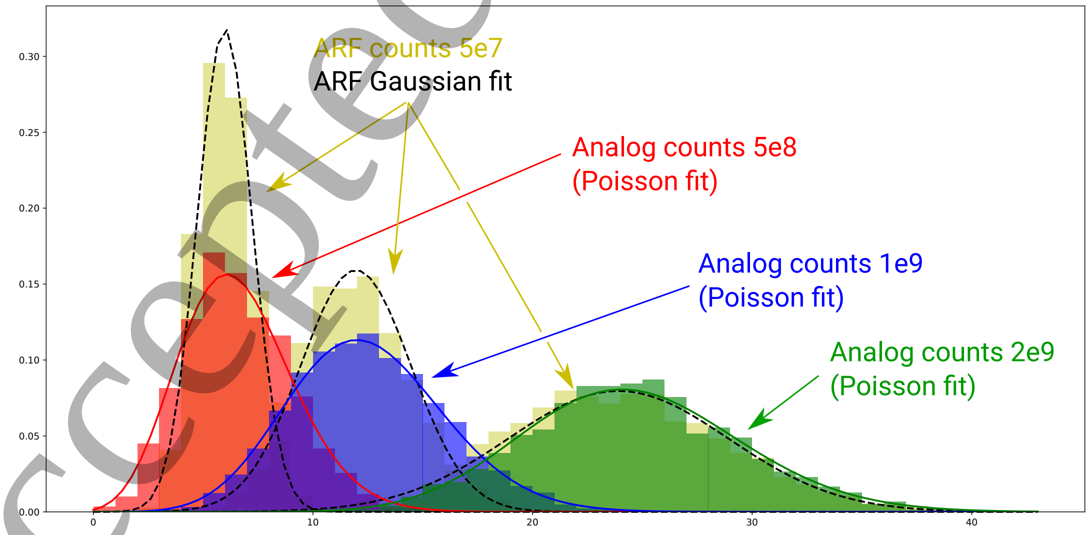

Figure 8. Count distributions in a homogeneous image area obtained from three analog Monte-Carlo simulations with increasing number of primary particles (5×10^8, 10^9 and 2×10^9), and for an ARF-nn simulation with 5×10^7 particles. Distributions from analog Monte-Carlo were fitted with a Poisson distribution, while distributions from ARF-nn were fitted with Gaussian (black dashed lines).

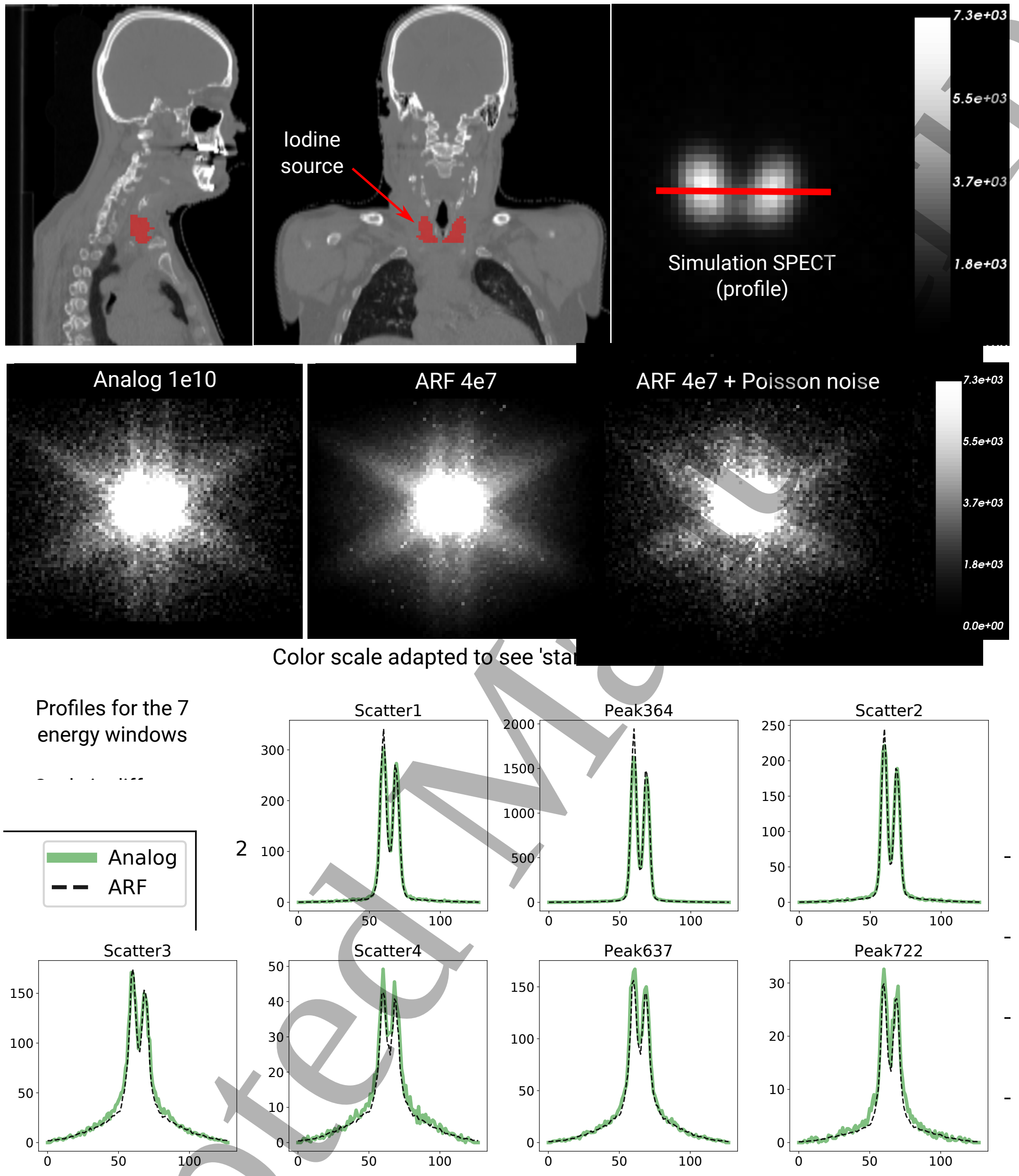

Figure 9. Complete SPECT simulation with patient CT and voxelized ^{131}I source. Top row images: CT image with source. Middle row images: SPECT images for the $W_{\rm Peak}$ 364 keV window: analog, ARF-nn, ARF-nn with Poisson noise. Note that the intensity scale is adapted to illustrate the 'star' effect. Bottom row: plots of SPECT profile for all energy windows comparing analog and ARF methods.

5. Discussion and conclusion

Figures 2, 3, 4 show that ARF-nn and ARF-histo are equally close to analog Monte-Carlo when the optimal binning is used. However, the binning parameters for which the best accuracy was achieved varied for each test simulation. For example, Test1 leads to best results with binning = 50, while for Test3 the best binning was 100. We also observed that ARF-nn lead to similar results for 10^9 to 10^8 events in the training dataset, while the results with ARF-histo degraded with decreasing dataset size. For example, Test1 required 10^9 primaries, while Test2 leads to correct results with 10^8 primaries, but using a different binning value. Figure 5 also illustrates that ARF-nn provides a better approximation than ARF-histo for the typical star effect of HEGP collimator. As a conclusion, ARF-nn requires less training data and provides more consistent results than ARF-histo in all evaluated cases.

It is shown that the ARF count distribution is Gaussian and that Poisson noise should be added to retrieve similar noise properties as with analog Monte-Carlo. We also show (figure 8) that mean count differ very little between analog and ARF (less than 1%). In figure 9, image profiles showed some differences that may partly be attributed to the stochastic nature of the process. The ARF image also tends to depict slightly higher count values compared to the analog image near high gradient areas. The origin of this behavior may be linked to the approximation made by ARF regarding the photon's impact location on the detector as shown in [6].

The efficiency of the ARF methods was evaluated per pixel, for three different test cases. Speedup compared to analog Monte-Carlo laid between 10 to 3000 and was similar for ARF-nn and ARF-histo. We showed that the ARF methods are more efficient for low count areas (speedup of 1000-3000) than for high count areas (speedup of 20-300). The mean efficiency depends on the configuration of the simulation and could not be generalized. Figures 6 and 7 illustrated the differences in the uncertainty distributions between analog and ARF simulations, showing an inverse behavior: Analog is slow to converge in low event areas while ARF is faster in those regions. Hence, the efficiency is higher for simulations with a large number of scattered photons such as with ^{131}I than for simulations with ^{111}In. For a complete SPECT simulation with photons tracked inside a patient CT, the overall speedup was between 80 and 400. As a practical example, an ARF image obtained in 2.5 hours of CPU time with 4×10^7 photons and an analog image obtained in 25 days CPU time with 10^{10} photons were visually almost indistinguishable (even in image areas with a low number of counts as illustrated figure 9).

To conclude, the proposed method is an alternative to the ARF-histo method that provides more consistent results and requires less training data. ARF efficiency has been characterized. The principle to learn a detector response with a neural network is general and may be extended to characterize the response function of other types of detector response systems.

Acknowledgments

This work was performed within the framework of the SIRIC LYriCAN Grant INCa-INSERM-DGOS-12563, and the LABEX PRIMES (ANR-11-LABX-0063) of Université de Lyon, within the program Investissements d'Avenir (ANR- 11-IDEX-0007) operated by the ANR. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] Ljungberg M. and Strand S.E. "A Monte Carlo program for the simulation of scintillation camera characteristics." *Computer methods and programs in biomedicine*, 29:257–272. ISSN 0169-2607. August 1989.

[2] Roshan H.R., Mahmoudian B., Gharepapagh E., Azarm A. and Islamian J.P. "Collimator and energy window optimization for 90 Y bremsstrahlung SPECT imaging: A SIMIND Monte Carlo study". *Applied Radiation and Isotopes*, 108:124–128. 2016.

[3] Cajgfinger T., Rit S., Létang J., Halty A. and D. S. "Fixed Forced Detection for fast SPECT Monte-Carlo simulation". *Physics in medicine and biology*, 63(5). 2018.

[4] Garcia M.P., Bert J., Benoit D., Bardiès M. and Visvikis D. "Accelerated GPU based SPECT Monte Carlo simulations." *Phys Med Biol*, 61(11):4001–4018. Jun 2016.

[5] Rydéen T., Heydorn Lagerlof J., Hemmingsson J., Marin I., Svensson J., Baath M., Gjertsson P. and Bernhardt P. "Fast GPU-based Monte Carlo code for SPECT/CT reconstructions generates improved 177Lu images". *European Journal of Nuclear Medicine and Molecular Imaging*, 5(1). 2018.

[6] Song X., Segars W.P., Du Y., Tsui B.M.W. and Frey E.C. "Fast modelling of the collimator-detector response in Monte Carlo simulation of SPECT imaging using the angular response function." *Physics in medicine and biology*, 50:1791–1804. ISSN 0031-9155. April 2005.

[7] Descourt P., Carlier T., Du Y., Song X., Buvat I., Frey E., Bardies M., Tsui B. and Visvikis D. "Implementation of angular response function modeling in SPECT simulations with GATE". *Physics in medicine and biology*, 55(9):N253. 2010.

[8] Sarrut D., Bardiès M., Boussion N., Freud N., Jan S., Létang J.M., Loudos G., Maigne L., Marcatili S., Mauxion T., Papadimitroulas P., Perrot Y., Pietrzyk U., Robert C., Schaart D.R., Visvikis D. and Buvat I. "A review of the use and potential of the GATE Monte Carlo simulation code for radiation therapy and dosimetry applications." *Med Phys*, 41(6):064301. Jun 2014.

[9] Allison J., Amako K., Apostolakis J., Araujo H., Arce Dubois P., Asai M., Barrand G., Capra R., Chauvie S., Chytracek R., Cirrone G., Cooperman G., Cosmo G., Cuttone G., Daquino G., Donszelmann M., Dressel M., Folger G., Foppiano F., Generowicz J., Grichine V., Guatelli S., Gumplinger P., Heikkinen a., Hrivnacova I., Howard a., Incerti S., Ivanchenko V., Johnson T., Jones F., Koi T., Kokoulin R., Kossov M., Kurashige H., Lara V., Larsson S., Lei F., Link O., Longo F., Maire M., Mantero a., Mascialino B., McLaren I., Mendez Lorenzo P., Minamimoto K., Murakami K., Nieminen P., Pandola L., Parlati S., Peralta L., Perl J., Pfeiffer a., Pia M., Ribon a., Rodrigues P., Russo G., Sadilov S., Santin G., Sasaki T., Smith D., Starkov N., Tanaka S., Tcherniaev E., Tome B., Trindade a., Truscott P., Urban L., Verderi M., Walkden a., Wellisch J., Williams D., Wright D. and Yoshida H. "Geant4 developments and applications". *IEEE Transactions on Nuclear Science*, 53(1):270–278. February 2006.

[10] Kingma D.P. and Ba J. "Adam: A method for stochastic optimization". *arXiv preprint arXiv:1412.6980*. 2014.

[11] Paszke A., Gross S., Chintala S., Chanan G., Yang E., DeVito Z., Lin Z., Desmaison A., Antiga L. and Lerer A. "Automatic differentiation in PyTorch". 2017.

[12] Assié K., Gardin I., Véra P. and Buvat I. "Validation of the Monte Carlo simulator GATE for indium-111 imaging." *Phys Med Biol*, 50(13):3113–3125. Jul 2005.

[13] Chetty I.J., Rosu M., Kessler M.L., Fraass B.A., Ten Haken R.K., Kong F.M.S. and McShan D.L. "Reporting and analyzing statistical uncertainties in Monte Carlo-based treatment planning." *Int J Radiat Oncol Biol Phys*, 65(4):1249–1259. Jul 2006.