PDE and Jitter modelling of SPAD Devices.

Rémi Helleboid[1], Denis Rideau[1], Norbert Moussy[2], Olivier Saxod[2], Jeremy Grebot[1], Isobel Nicholson[1], Antonin Zimmerman[1], and Sara Pellegrini[1]

[1] *ST Microelectronics, Crolles, France*
[2] *CEA LETI, Grenoble, France*

Abstract

In this paper we present a full 3D simulation methodology to extract Photon Detection Probability (PDP) and Jitter of Single-Photon Avalanche Diode (SPAD) Devices. The simulation results are compared with measurements on devices and show good agreement with the experiments.

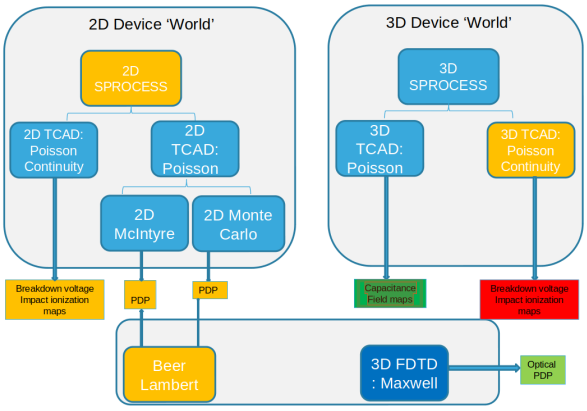
***Keywords*—** single-photon avalanche diode (SPAD), photon detection probability (PDP), jitter, avalanche breakdown probability, breakdown voltage

1 Introduction

Single Photon Avalanche Diodes (SPAD) are key optoelectronic detectors for medical imaging, camera ranging and automotive laser imaging detection and ranging (LiDAR) applications. Currently, the device leading the market is a micrometric silicon (Si) PN junction associated to a proximity CMOS electronics biasing the system above the breakdown voltage. Si-SPADs present low noise and relatively high photon detection probability (PDP), but their sensitivity is limited to photon wavelengths lower than 1100 nm, while class 1 eye-safety devices would require wavelengths larger than 1400 nm.

2 Device structure and TCAD simulation

Figure 1: SPAD simulation workflow



3 Avalanche breakdown probability

The avalanche breakdown probability is computed by the means of the well known McIntyre model [Oldham et al., 1972]. We briefly recall the model derivation : Let $P_e(x)$ be the probability that an electron starting at x in the depletion layer triggers an avalanche and $P_h(x)$ the same probability for an hole starting at x. Straightforwardly, the probability that neither an hole nor an electron starting at x trigger an avalanche is given by $(1-P_e(x))(1-P_h(x))$. Thus, the probability that either the hole or the electron trigger an avalanche, noted P_{pair} is :

$$\begin{aligned} P_{pair}(x) &= 1-(1-P_e(x))\left(1-P_h(x)\right) \\ &= P_e + P_h - P_e P_h \end{aligned}$$

Now, the probability that an electron starting at $x+dx$ triggers an avalanche is : The probability that the electron reaches the position x and triggers an avalanche in x plus the probability that it triggers an avalanche between x and $x+dx$ less the probability of the intersection of the two previous events. It writes :

$$\begin{aligned} P_e(x+dx) &= P_e(x) + \alpha_e(x)dxP_{pair}(x) - P_e(x)\alpha_e dxP_{pair}(x) \\ &= P_e(x) + \alpha_e(x)dx(P_e(x)+P_h(x)-P_e(x)P_h(x)) \\ &\quad - P_e(x)\alpha_e(x)dx(P_e(x)+P_h(x)-P_e(x)P_h(x)) \\ &= P_e(x) + dx\,\alpha_e(x)(P_e(x)+P_h(x)-P_e(x)P_h(x))(1-P_e(x)) \end{aligned}$$

Where α_e is the electron linear ionization rate : the probability by length that an electron create an impact ionization event.
One can rearrange the terms to obtain :

$$\frac{P_e(x+dx)-P_e(x)}{dx} = \alpha_e(x)(P_e(x)+P_h(x)-P_e(x)P_h(x))(1-P_e(x))$$

Which leads to the first ordinary differential equation :

$$\frac{dP_e}{dx} = (1-P_e)\alpha_e(P_e+P_h-P_eP_h)$$

The same reasoning applies to the probability that an hole starting at $x-dx$ triggers an avalanche. Which leads to the second ordinary differential equation :

$$\frac{dP_h}{dx} = -(1-P_h)\alpha_h(P_e+P_h-P_eP_h)$$

Therefore we can draw up the McIntyre system :

$$\begin{cases} \dfrac{dP_e}{dx} = (1-P_e)\alpha_e(P_e+P_h-P_eP_h) & (1) \\ \dfrac{dP_h}{dx} = -(1-P_h)\alpha_h(P_e+P_h-P_eP_h) & (2) \end{cases}$$

for $0 \le x \le W$.
Adding the couple of boundary value conditions :

$$\begin{cases} P_e(x=0) = 0 & (3) \\ P_h(x=W) = 0 & (4) \end{cases}$$

we have a full 1D coupled and non-linear boundary value problem. Since we have to extract this value at a large number of points, we use a self-made solver, embedded in a C++ program. This solver uses finite difference method coupled with a Newton's method to care of the non-linearity of the problem [Ascher et al., 1987]. The algorithm is different from those implemented in MatLab routine (bvp4c) or SciPy function (solve_bvp) [Kierzenka and Shampine, 2001] but the comparison with these tools show no difference.
We set the following notations :

$$Y(x) = \begin{pmatrix} P_e(x) \\ P_h(x) \end{pmatrix}$$

$$f(Y,x) = \begin{pmatrix} (1-Y_1(x))\alpha_e(Y_1(x)+Y_2(x)-Y_1(x)Y_2(x)) \\ -(1-Y_2(x))\alpha_h(Y_1(x)+Y_2(x)-Y_1(x)Y_2(x)) \end{pmatrix}$$

$$g(s_1,s_2) = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$

The problem hence reads :

$$\begin{cases} Y'(x) = f(Y,x) & (5) \\ g(Y(0),Y(w)) = 0 & (6) \end{cases}$$

3.1 Newton scheme

3.1.1 Finite differences approximation

Let $\mathcal{M}$ be the mesh on which we work. It is given by the streamlines construction. So we have :

$$\mathcal{M} : 0 = x_1 < x_2 < x_3 < \cdots < x_N < x_{N+1} = w$$

The approximated solution on mesh $\mathcal{M}$ is $Y_{\mathcal{M}} = (y_1, y_2, \cdots, y_N, y_{N+1})$, where y_i is the approximation of $Y(x_i)$.
For numerical approximation we again consider the mesh $\mathcal{M}$ and denote the vector of approximate solution values at mesh points by $Y_{\mathcal{M}}$. The trapezoidal scheme of finite difference methods is given by :

$$\frac{y_{i+1}-y_i}{h_i} = \frac{1}{2}\left(f(x_{i+1},y_{i+1}) + f(x_i,y_i)\right) \quad 1 \le i \le N \qquad (7)$$

$$g(y_1, y_{N+1}) = 0 \qquad (8)$$

Thus we obtain a system of $2(N+1)$ algebraic equations for the $2(N+1)$ unknowns $Y_{\mathcal{M}}$. Unlike before, though, these equations are non-linear. The number N depends on the precision we take when we construct the streamlines. We commonly take a range of $[1nm, 10nm]$, we then have $N \sim 5000$.
Fortunately, the Jacobian matrix of this system is rather sparse, as we shall see below.

3.1.2 Newton method

We consider a system of equation written in the compact form :

$$\mathbf{F(s) = 0}$$

We define a function $\mathbf{G}$:

$$\mathbf{G(s)} = \mathbf{s} - \left[\mathbf{F}'(\mathbf{s})\right]^{-1}\mathbf{F(s)}$$

with $\mathbf{F}'(\mathbf{s})^{-1}$ the inverse of the Jacobian matrix of F :

$$\mathbf{F}'(\mathbf{s}) = \frac{\partial \mathbf{F(s)}}{\partial \mathbf{s}}$$

Then the newton method iterative method is given by the iteration :

$$\mathbf{s}^{k+1} = \mathbf{G}(\mathbf{s}^k)$$

So the algorithm will first solve the linear system :

$$\mathbf{F}'\left(\mathbf{s}^k\right)\boldsymbol{\xi} = -\mathbf{F}\left(\mathbf{s}^k\right) \qquad (9)$$

And then simply do :

$$\mathbf{s}^{k+1} = \mathbf{s}^k + \boldsymbol{\xi} \qquad (10)$$

3.1.3 Construction of the linear system

Let $\mathbf{N}_{\mathcal{M}}$ be the following discrete differential operator :

$$\mathbf{N}_{\mathcal{M}}\mathbf{y}_i = \frac{y_{i+1}-y_i}{h_i} - \frac{1}{2}\left(f(x_{i+1},y_{i+1}) + f(x_i,y_i)\right)$$

Then

$$\mathbf{F(s)} = \begin{pmatrix} N_{\mathcal{M}}\mathbf{y}_1 \\ N_{\mathcal{M}}\mathbf{y}_1 \\ \vdots \\ N_{\mathcal{M}}\mathbf{y}_N \\ g(y_1, y_{N+1}) \end{pmatrix}$$

We set

$$\boldsymbol{\xi} = \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_N \\ \mathbf{w}_{N+1} \end{pmatrix}$$

So that the Newton method iteration becomes :

$$\frac{\mathbf{w}_{i+1}-\mathbf{w}_i}{h_i} - \frac{1}{2}\left[A(x_{i+1})\mathbf{w}_{i+1} + A(x_i)\mathbf{w}_i\right] = -\mathbf{N}_{\mathcal{M}}\mathbf{y}_i^k \quad 1 \le i \le N \qquad (11)$$

$$B_a w_1 + B_b w_{N+1} = -g\left(y_1^m, y_{N+1}^m\right) \qquad (12)$$

Where A is the following matrix :

$$A\left(x_j\right) := \frac{\partial f}{\partial y}\left(x_j, \mathbf{y}_j^k\right)$$

And with

$$B_a = \frac{\partial g(\mathbf{y_1^k}, \mathbf{y_{N+1}^k})}{\partial \mathbf{u}}, \quad B_b = \frac{\partial \mathbf{g}(\mathbf{y_1^k}, \mathbf{y_{N+1}^k})}{\partial \mathbf{v}}$$

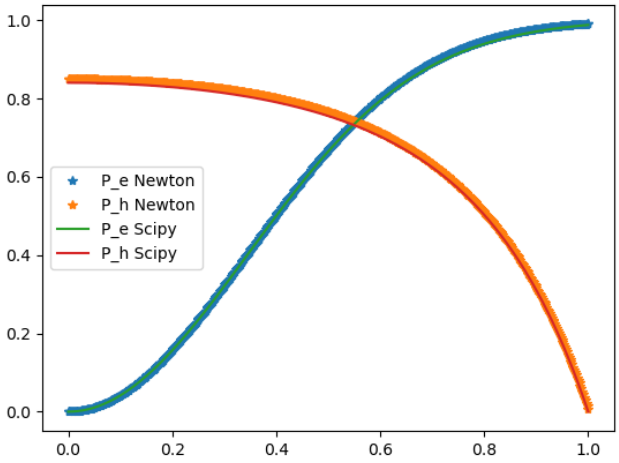
which here turns into :

$$B_a = 1, \quad B_b = 1$$

3.1.4 Final algorithm and comparison with SciPy function

The corresponding algorithm is details at algorithm 1. The algorithm results were checked with the SciPy routine as a reference. When the number of points is greater than 500, the relative error is always bellow 1%.

Figure 2: Comparison of the Newton's method agaisnt the SciPy routine with 512 points.



3.2 Application to field lines

The McIntyre model is a fully 1D model where the electron and holes path are assumed to be a straight line, often took from the bottom to the top of the device. In this work we wish to have accurate values of breakdown probability in all the device volume. To this purpose we use electric field streamline to model the carriers transport inside the device. While this modelization might be less accurate than a drift-diffusion model, it would not be consistent with the McIntyre model, where electrons and holes are assumed to follow the same path. The field lines are computed straightforwardly using simple Euler scheme with adaptive step to ensure a good distribution of points along the line.

$$\frac{dX(s)}{ds} = \vec{F}_{electric} \qquad (13)$$

Algorithm 1: Newton's Method Solver for BVP

input : The Boundary value Problem
input : The Mesh $\mathcal{M}$
input : An initial guess of $Y_{\mathcal{M}}$
input : A maximal tolerance TOL
input : A maximal number of iterations
output: The Solution $Y_{\mathcal{M}}$
output: The final residual error

RES $\leftarrow$ 1000 ;
$NbIterations \leftarrow 0$;
Initialize $w_{\mathcal{M}}$ as a vector of size 2N;

while $RES > TOL$ *and* $NbIterations < MaxNbIterations$ **do**
  for *i=1 to 2N* **do**
    Construct S_i;
    Construct R_i;
    Construct $q_i = -\mathbf{N}_{\mathcal{M}} y_i$;
  Construct A;
  Construct β;
  Solve $Aw_{\mathcal{M}} = \hat{\beta}$;
  for *i=1 to 2N* **do**
    $y_i \leftarrow y_i + (w_{\mathcal{M}})_i$
  $RES \leftarrow ||w_{\mathcal{M}}||$;
  $NbIterations \leftarrow NbIterations + 1$;
if $RES \leq TOL$ **then**
  //The method has converged ;
  return Y;
else
  //The method has converged ;
  return Error : No Convergence

Then the streamline is

$$\{X(s) \text{ for } s \in [s_0, s_f]\}$$

Our Euler method then reads :

$$\frac{X(s+ds) - X(s)}{ds} = \vec{F}_{electric}(X(s)) \tag{14}$$

$$\implies X(s+ds) = X(s) + \underbrace{ds\vec{F}_{electric}(X(s))}_{dX} \tag{15}$$

So with the discretization, calling X^k the approximation of $X(k*ds)$ we have :

$$\frac{X^{k+1} - X^k}{ds} = \vec{F}_{electric}(X^k) \tag{16}$$

$$\implies X^{k+1} = X^k + \underbrace{ds\vec{F}_{electric}(X^k)}_{dX^k} \tag{17}$$

This operation is computed both forward (hole motion) and backward (electron motion). We then interpolate the electric field on the line. We set one extremity of the line as it's beginning and the other end as it's end. We can now obtain a function E(x) where x is the distance from the beginning of the line and E(x) is the norm of the electric field at this point. The streamline and this function are represented in figures 4 and 3.
We can now compute the impact ionization coefficients, requirend to compute the McIntyre's model, in this work we choose the local coefficient from Van Overstraeten and De Man [Van Overstraeten and De Man, 1970].
We can compute the breakdown probability over multiple streamlines starting from multiple points inside the device and plot them to have an idea of the breakdown probability inside the device, see figure .

Figure 3: Field line of electric field inside the device

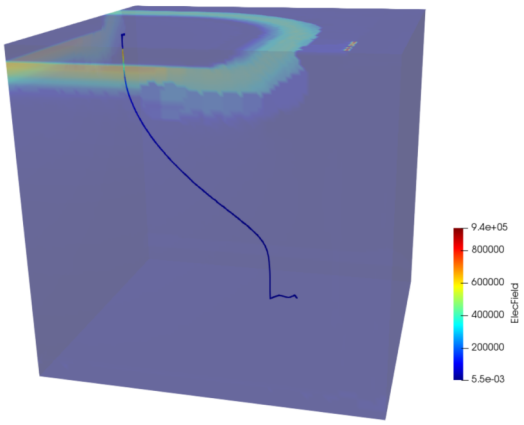
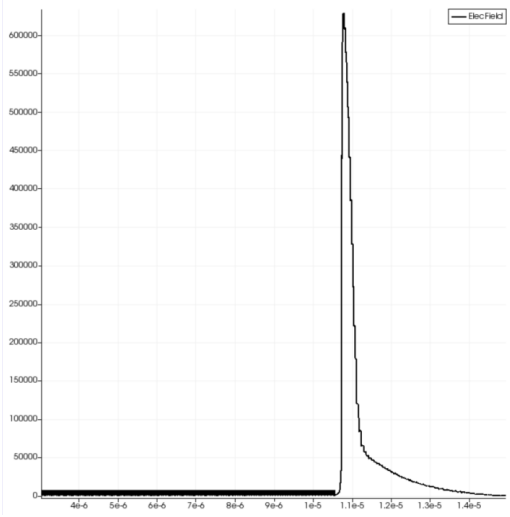


Figure 4: Plot of the electric field norm along the field line



4 Jitter modeling

4.1 Drif-Diffusion model

The jitter in SPAD devices is made of multiple phenomena such as depth of photon absorption, carrier transport timing, avalanche build-up,quench circuit statistics etc.
In the present work, we focus on the modeling of the carrier transport part. To do so, we put together the streamlines and the advection-diffusion equation with variable velocity and diffusion. Let us consider that a photon absorption at a given point x_0 leads to the creation of an electron-hole pair, we want to simulate the timing for the electron to reach the avalanche zone, we assume that this zone is represented by the location of the maximum electric field, denoted $x_{E_{max}}$. Let $f : (x,t) \mapsto f(x,t)$ the probability distribution of an electron presence. We assume that the electron will drift along the electric field streamline. At time $t = 0$, f is a Dirac distribution, in order to be able to compute a solution numerically, we start a time $t_0 = \delta t$ where δt is as small as possible. We then take the assumption that $D(x) = D(x_0)$ and $u(x) = u_(x_0)$, the analytical solution of the advection-diffusion applies.
Hence, f verify the following equation :

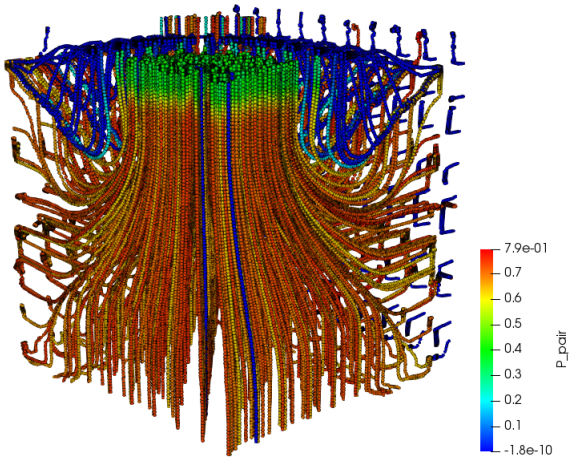
$$\forall x \in [x_s, x_{max}] \text{ and } t \in [t_0, T]$$

$$\frac{\partial f}{\partial t}(x,t) = -\frac{\partial (u \cdot f)}{\partial x}(x,t) + \frac{\partial}{\partial x}\left(D \cdot \frac{\partial f}{\partial x}\right)(x,t) \tag{18}$$

With the following initial condition :

$$f(x, t = t_0) = \frac{1}{\sqrt{4\pi D(x_0) t_0}} exp(-\frac{(x - v(x_0) t_0)^2}{4D(x_0) t_0} \tag{19}$$

Figure 5: Breakdown Probability computed over multiple streamlines



And setting an absorbing boundary condition at $x_s = x_{E_{max}}$:

$$\frac{\partial f}{\partial t}(x_s,t) = \frac{\partial}{\partial x}\left(D \cdot \frac{\partial (f)}{\partial x}\right)(x_s,t) \tag{20}$$

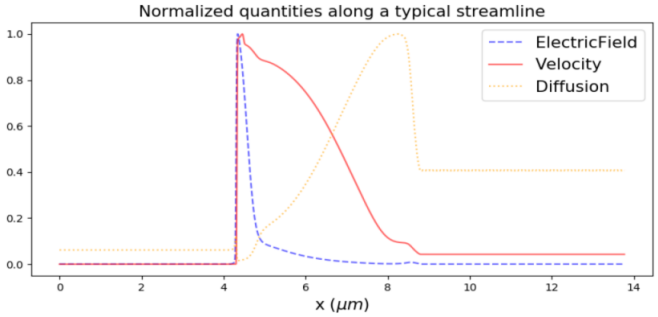
The velocity is computed throw a high field saturation model and the diffusion throw the Einstein relation $D = \frac{\mu k_B T}{q}$, these quantities are represented in figure 6. Let T_e the time for the electron to reach the avalanche region. It is straightforward that the probability that T is less than t is:

$$P_r(t<T) = F_{Pr}(t) = 1 - \int_{x_0}^{x_{max}} f(x,t)dx$$

From this cumulative distribution function, we can find back the distribution of T:

$$f_T(t) = \frac{F_{Pr}(t)}{dt} \tag{21}$$

Figure 6: Normalized data along a streamline



4.2 Numerical solution

First, we discretize the streamline Ω as

$$\Omega_h = \{x_s = x_0, x_1, x_2, \cdots, x_{N-1}, x_N = x_{max}\}$$

And time interval as :

$$[t_0, t_{max}] = \{t_0 = t_0, t_1, t_2, \cdots, t_{M-1}, t_M = t_{max}\}$$

The approximation of the solution is noted :

$$f(x_i,t_k) \simeq f_i^k \quad \text{for } i \in \{0,\cdots,N\} \text{ and } k \in \{0,\cdots,M\}$$

The equation is solved by the mean of the finite difference method, more precisely we use a modified Crank-Nicholson method. The differentiation of the equation reads :

$$\begin{aligned}\frac{f_i^{k+1}-f_i^k}{dt} =& \frac{1}{2}\left[-u_i\frac{f_{i+1}^{k+1}-f_{i-1}^{k+1}}{2h} + D_i\frac{f_{i+1}^{k+1}-2f_i^{k+1}+f_{i-1}^{k+1}}{h^2}\right] \\ &+\frac{1}{2}\left[-f_i^k\frac{u_{i+1}-u_{i-1}}{2h} + \frac{D_{i+1}-D_{i-1}}{2h}\frac{f_{i+1}^{k+1}-f_{i-1}^{k+1}}{2h}\right] \\ &+\frac{1}{2}\left[-u_i\frac{f_{i+1}^{k}-f_{i-1}^{k}}{2h} + D_i\frac{f_{i+1}^{k}-2f_i^{k}+f_{i-1}^{k}}{h^2}\right] \\ &+\frac{1}{2}\left[-f_i^k\frac{u_{i+1}-u_{i-1}}{2h} + \frac{D_{i+1}-D_{i-1}}{2h}\frac{f_{i+1}^{k}-f_{i-1}^{k}}{2h}\right]\end{aligned}$$

It is then straightforward that the resulting scheme will be a linear system to solve with a matrix vector product as right hand side member:

$$AF^{k+1} = BF^k$$

The corresponding algorithm would then be :

Algorithm 2: Crank Nicolson algorithm

input : The initial condition
input : The space step h and time step dt
input : The times t_0 and t_{max}
output: The Solution F^M

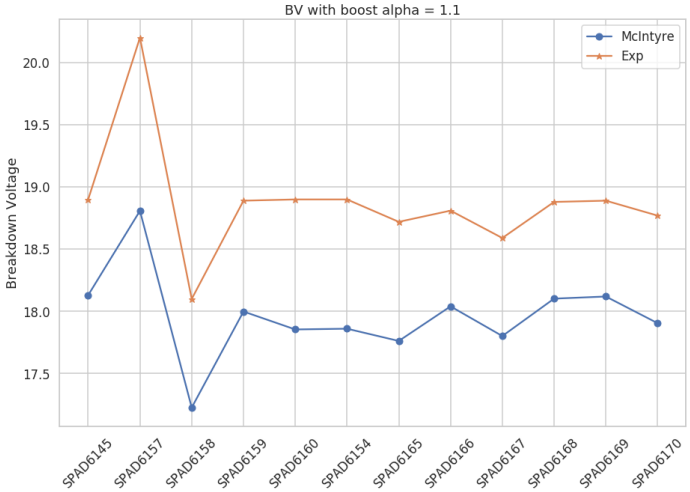
time $\leftarrow t_0$;
Initialize F as a vector of size N with finitial values;
Construct the matrix A and B defined previously.

while $time <= t_{max}$ **do**
  $Y \leftarrow BF$
  Solve $AX = Y$
  $F \leftarrow X$
  $time \leftarrow time + dt$

return F

5 Results and comparisons with experiments

Figure 7: Comparison of Breakdown Voltage obtained by McIntyre model versus experiments for multiple SPAD designs



6 Discussion

References

[Ascher et al., 1987] Ascher, U. M., Mattheij, R. M. M., and Russell, R. D. (1987). *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Society for Industrial and Applied Mathematics, Philadelphia, 1st edition edition.

[Gulinatti et al., 2009] Gulinatti, A., Rech, I., Assanelli, M., Ghioni, M., and Cova, S. D. (2009). Design-oriented simulation of the Photon Detection Efficiency and temporal response of Single Photon Avalanche Diodes. In *2009 IEEE LEOS Annual Meeting Conference Proceedings*, pages 297–298, Belek-Antalya, Turkey. IEEE.

[Hill and Miller,] Hill, T. P. and Miller, J. How to Combine Independent Data Sets for the Same Quantity. page 20.

[Karahan, 2007] Karahan, H. (2007). Unconditional stable explicit finite difference technique for the advection–diffusion equation using spreadsheets. *Advances in Engineering Software*, 38(2):80–86.

[Kierzenka and Shampine, 2001] Kierzenka, J. and Shampine, L. F. (2001). A BVP solver based on residual control and the Maltab PSE. *ACM Transactions on Mathematical Software*, 27(3):299–316.

[Oldham et al., 1972] Oldham, W., Samuelson, R., and Antognetti, P. (1972). Triggering phenomena in avalanche diodes. *IEEE Transactions on Electron Devices*, 19(9):1056–1060.

[Pancheri et al., 2014] Pancheri, L., Stoppa, D., and Dalla Betta, G.-F. (2014). Characterization and Modeling of Breakdown Probability in Sub-Micrometer CMOS SPADs. *IEEE Journal of Selected Topics in Quantum Electronics*, 20(6):328–335.

[Spinelli and Lacaita, 1997] Spinelli, A. and Lacaita, A. L. (1997). Physics and numerical simulation of single photon avalanche diodes. *IEEE Transactions on Electron Devices*, 44(11):1931–1943. Conference Name: IEEE Transactions on Electron Devices.

[Sun et al., 2019] Sun, F., Xu, Y., Wu, Z., and Zhang, J. (2019). A Simple Analytic Modeling Method for SPAD Timing Jitter Prediction. *IEEE Journal of the Electron Devices Society*, 7:261–267.

[Van Overstraeten and De Man, 1970] Van Overstraeten, R. and De Man, H. (1970). Measurement of the ionization rates in diffused silicon p-n junctions. *Solid-State Electronics*, 13(5):583–608.