ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di: LNF - INFN

SPARC/EBD-07/002
14 Giugno 2007

A robust algorithm for beam emittance and trace space evolution reconstruction

Daniele Filippetto,
INFN, Laboratori Nazionali di Frascati (LNF)

Abstract

A robust algorithm for image analysis and emittance calculation from 1D pepper-pot measurements is presented here. The need of a robust and consistent algorithm increases when not only the beam parameters at a fixed point are measured, but the evolution of such parameters along the direction of beam propagation. In this case indeed, the extracted values have to be compared one with each other and, since the parameter variations and oscillations can be very small, one has to make sure that these variations do not come from different signal treatments, but they have a physical meaning. The program is divided in 3 main routines: the image analysis, the data cleaning, and the trace space reconstruction. Each of these routines is described in details together with the choices done at any single step. Some tests on the image analysis using analytical profiles are also presented, to better understand the limitations of the algorithm. Used thresholds are physically explained, and an overview of the used procedure to reconstruct the trace space from the data is given.

Published by Laboratori Nazionali di Frascati

1 Introduction to the method of measurement

The Emittance-meter is a novel diagnostic used to measure the transverse emittance along 2 meters in the cathode vicinity. The beam in this region is quasi-relativistic and the classic method such as quadrupole scan can not be used since for a fixed beam energy and electron density internal collective forces (space charge) play a dominant rule in the beam transport. The classic matrix formalism can't be applied unless a matrix formalization for the "space charge kick" and a distributed model are introduced ([4]). The alternative way is to build a system where the beam, starting from a fixed position, is no longer subject to internal forces, decreasing its current as much as needed to make beam evolution dominated by initial temperature from that position on. The easiest way to make it is using a collimator that stops all particles but that fitting trough it, decreasing its current. The exit beamlet is no longer *"space charge dominated"* and, giving it the time to evolve over a certain distance, one can observe light emitted from a scintillation screen hit by the beam. If the collimator is a slit, it cuts the beam only along a specified axis (X or Y); moving the slit in different positions along that axis, the beam is sampled. From the beamlet image, a profile along the sampling direction is obtained, integrating the image over the other axis. Analyzing the profile the beamlet initial (at the slit position) mean divergence and the velocity dispersion along the sampling direction are deducted. By doing this for different slit positions, and correlating position and divergence, the $1D$ emittance can be calculated, and phase space reconstructed. A novel device making use of this principle has been built at SPARC photo-injector, in order to better understand, control and manipulate electron beam created by the laser-cathode-gun system, and calculate the maximum exit transverse brilliance.

Since the beam properties have a big range of possible values, a flexible device is needed, in which all the measurement significant distances, such as the step between one slit and the following or the distance between the slit and the screen, can be changed any time. The single slit method is used, where a single slit moves along the beam transverse dimension making several samples in different positions. The number of samples and the relative distance are free parameters. Single slit method has the drawback to be a multi-shot measure, opposite to the single-shot multi slit method, in which however the flexibility is much less (the number of slit and their relative distances are fixed parameters this time).

The measurement procedure consists on beam tracing along the measurement region via an envelope scan first, calculating the beam centroid and rms dimensions in a variable number of points. In this first step the slit mask is extracted and images are taken looking to the entire beam on the Yag:Ce screen via a $1 : 1.5$ imaging optics. Then an

automatic emittance scan measurement starts, making use of previously calculated parameters to center the slit and fix the slit step in different positions.

The slit method is well known in accelerator beam physics and diagnostics, and lots of references are available in literature. We refer to ref.[2] for details. In the following a qualitative understanding of the method is given. Starting from the end, the RMS formula for emittance calculation (see for example ref.[5])

$$\varepsilon_{nx} = \beta\gamma\sqrt{\langle x^2\rangle \langle x'^2\rangle - \langle xx'\rangle^2} \tag{1}$$

where β and γ are the Einstein coefficients, x is the spatial coordinate, and $x^{'}$ is the derivative of x versus z, i.e. the angle of the velocity vector respect to the ideal trajectory (z). The formula already gives us indications on what beam parameters need to be measured (space, angles and energy)and what are problems with emittance measurements: overall it is visible that emittance is calculated from the difference of two big numbers (order of 10^{-6}). This difference has to give results of the order of 10^{-12}, meaning that the two quantities inside the square root need to be calculated very precisely.
While it is easy to extrapolate the beam position and dimensions at any place, even with self forces acting on the beam, to calculate the divergence in one place one need two measurement points. Independently from the quantities measured at each point (that can be magnetic strength or beam dimension, depending on the particular measurement), the hypothesis is that the beam evolves in between just under the influence of measured initial quantity and, in case, some other externally applied forces. This hypothesis is no more valid inside a "quasi relativistic" dense beam, where also self forces play a rule. In this case the evolution of the beam can't be described without knowing also the initial beam density and divergence, and without having a perfect model of intra-beam interactions. In such a beam the single particle divergence is changing from time to time, under the influence of self forces: this is definitely the cause of quad scan like measurements failure in SP dominated beams (ref.[6]). The position-angle correlation also is changing rapidly in a way not only dependent from initial conditions.

A MATLAB based software has been developed for image manipulation and parameter extraction. A lot of work has been done in image filtering and data analysis, in order to create a consistent and user-friendly program to calculate beam parameters from raw data. In following sections the main parts of the program will be explained separately, and the choices made to analyze data presented.

In fig.(1) is showed the entire program dataflow. I consists in 4 principal routine (the first four rows of dataflow in picture), followed by other additional routine for trace space movie and visualization (last two lines). These latter routines do not participate to the data analysis and emittance calculations.

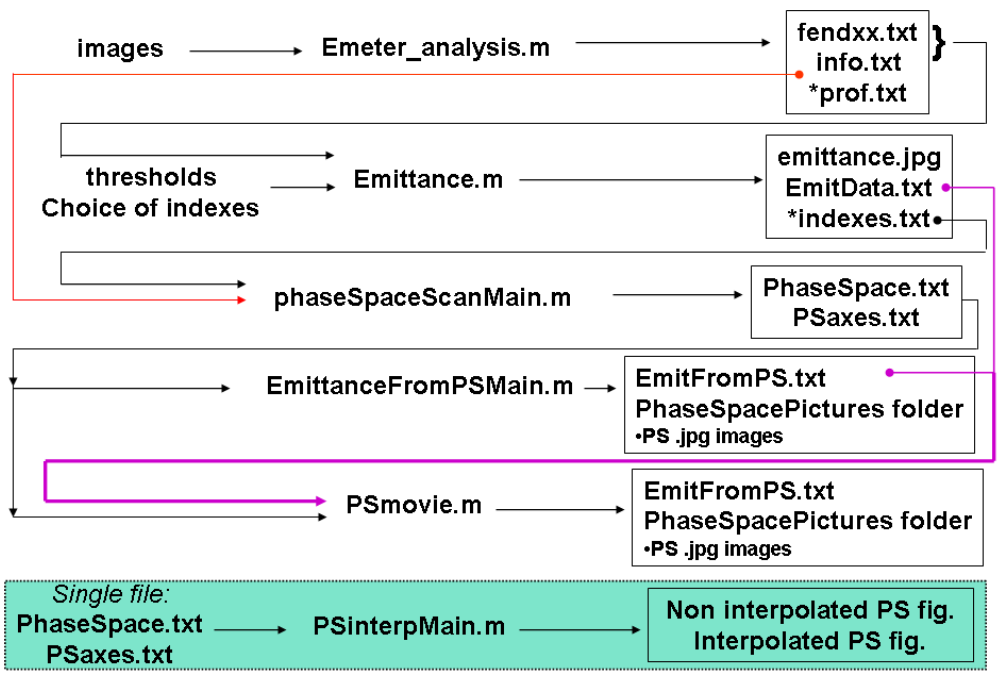

Figure 1: The analysis software data flow. To each line corresponds a separate routine;

In the central column of the picture the routine names are written; the input parameter of each routine are summarized on the left column, while the right part show the specific routine output. As can be seen from the picture, except for the first routine, all the others have input parameters depending from the previous, so that there is a precise order that has to be followed, to run the program. It is the order that goes from up to the bottom of the picture. The first routine takes as input the row images, and it gives as output $.txt$ files with parameters obtained from image analysis. This routine contains indeed the algorithm developed to find and denoise the signal, calculating the needed values. It takes time (about 40 minutes) to analyze an entire emittance scan (30 different emittance measurements, with 13 slit position per measure and about 30 images per slit position, leading to about 11700 images), while the others are much faster (less than a minute). This is the main reason that lead us to divide different logical routines in four parts, giving the possibility to run them independently.

The second routine takes as input the files saved by the first one and calculates the emittance in two different ways that will be explained exhaustively in section 3.
The third routine reconstruct and save in a $.txt$ file the trace spaces for each single emittance measurement, while the fourth interpolates these spaces to obtain more sampling

along the abscissa, and than calculates the emittance from the trace space, with the desired cut in charge (by default the 95% of the beam charge is used).

2 Single image analysis

Data filtering is probably the most delicate step in all the entire measurement process, the art to manipulate data stored and extrapolate from it the wanted parameters value cleverly distinguishing noise from signal, deciding what is useless and what fundamental. In our case, the data taken are beamlet images, and we would like to develop an automatic procedure (given the amount of data stored) to analyze them, choosing MATLAB as programming environment. The parameters to be extrapolated from image are: beamlet area (proportional to the number of particle passed trough to the slit), its centroid, and the rms width along the direction perpendicular to slit edges. A profile along that direction is indeed extracted form the image integration along the other axis. This profile is then manipulated to calculate the needed parameters. Furthermore, the knowledge of the slit center, the slit width, and the distance between sampling and imaging plane, permits the rms emittance calculation. Obviously to know the normalized emittance also the energy has to be measured. The formulas that we use to calculate profile centroid ad rms are the weighted means:

$$A = \sum_{i=1}^{n_j} a_i; \tag{2}$$

$$\bar{x} = \frac{1}{A} \sum_{i=1}^{n_j} a_i x_i; \tag{3}$$

$$\left\langle x^2 \right\rangle = \frac{1}{A} \sum_{i=1}^{N} a_i \left(x_i - \bar{x}\right)^2 = \frac{1}{N} \sum_{i=1}^{N} a_i x_i^2 - \bar{x}^2; \tag{4}$$

Where a_i is the profile value (y value in the example profile of fig.(2)) at i-th position, corresponding to the integrated i-th image line (or column), and x_i is the corresponding abscissa value (x axis in the example profile of fig.(2)).

A systematic study of images acquired has been done fixing the ranges of variability of parameters to be extrapolated and other important quantities:

- A baseline on the profile is always present, whose value depend on the camera gain, but for our typical measurement is always around 10^4 ;
- Noise level peak to peak is around 300 A.U.(depending from gain), randomly distributed on the baseline;
- The signal peak varies from 800 to 8000 A.U., depending from gain, but mainly from which part of the beam is sampled;
- The profile rms width is strongly related to the beamlet divergence at the slit plane. It is proportional to the uncorrelated spread in angles or, roughly speaking, to

the transverse trace space thickness at one fixed x value. Studying the different trace space reconstructed (the reconstruction method will be explained in section 4) from measurements, and looking to some selected "$extreme$" trace space plots, we found this spread to be between 0.2 and $1mrad$. If one consider the distance between slit and imaging plane equal to $400mm$, and the virtual dimensions of the camera pixel (physical dimension times the optics magnification), the expected rms width range goes from 5 to 20 pixels.

- The signal centroid can be everywhere inside the profile.

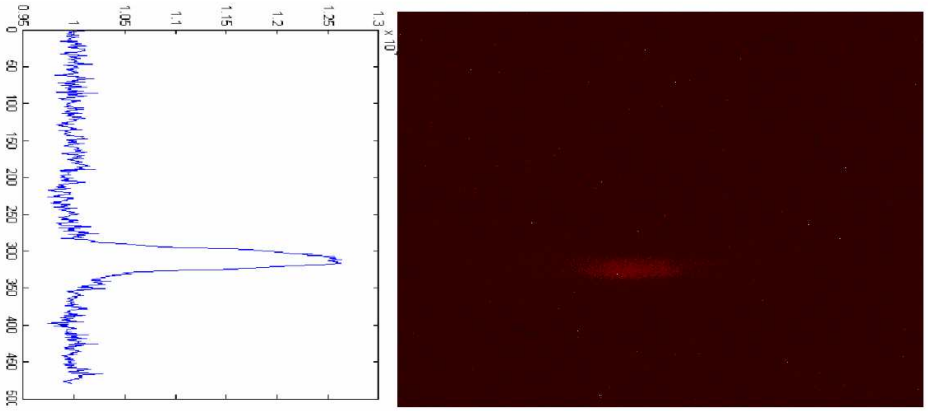

Figure 2: example of data; beamlet image and its profile.

For a particular measurement all the mentioned values have correlations, so that it is not obvious to associate for instance the smallest profile width with the highest profile height. If one has for example a *butterfly-like* transverse trace space, there is an inverse correlation between height and rms, i.e. the bigger the latter the smaller the former (see for example fig.(14). On the other end the correlations change for different measurements, and considering the entire data set, every combination is possible.

2.1 Signal detection

Once the profile is created from image integrating it along the direction normal to the slit motion, the first logical step (*signalDetect.m*) is to select a region of interest containing all the signal, and extract it. At this point we're not interested in finding exact output values for the signal, so that, even though signal shape changes completely between different measurements, the signal is fitted with a Gaussian-plus-baseline function:

$$f(x) = a \cdot e^{\frac{(x-c)^2}{2s^2}} + h. \tag{5}$$

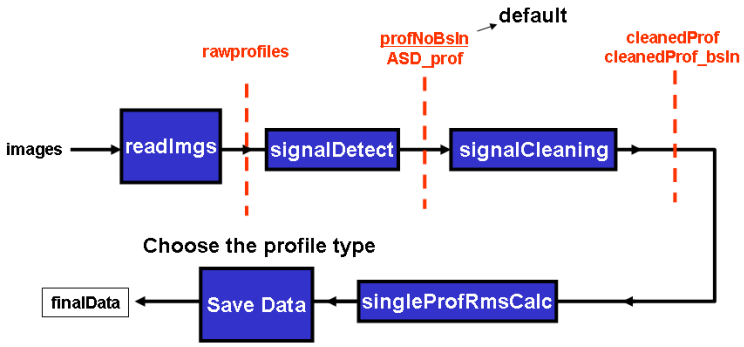

Figure 3: Analysis program dataflow; the names inside blue boxes are the code function names. Red dashed points are positions were the different variables are extracted and can be saved.

Initial fit parameters are calculated using as function height (a) the maximum of the smoothed profile (moving average with "10 pixel average") and as temporary centroid (c) its position; to calculate the initial baseline (h) and sigma (s), an initial window corresponding to a width that would have a signal with $4mrad$ angles spread (4 times the maximum value possible for our typical beam) centered around the maximum is taken, and the baseline is calculated averaging all the pixel out of window, while the initial rms width will be that of the signal inside the window. At the end the centroid is recalculated as the ROI-signal ($RegionOfInterest$) center of mass. This last action turned out to be very important, because, as we'll see later in this section (see for example fig.(4), the signal is not always symmetric respect to the maximum, and the center of mass can be slightly different from it; this operation allows to center the final ROI respect to the whole signal and not just to its peak. There's no *a priori* reason to give more importance to the stronger signal than to the weaker. From the fit one gets new rms, centroid, height and baseline of the profile. The original signal is now manipulated, subtracting the baseline and limiting the region of interest to ± 5 sigma around the centroid. The function outputs are:

1. the original signal without baseline (variable name inside the program: *ProfNoBsln*);

2. the new region of interest containing all the signal with the baseline subtracted (*ASD_prof* , After Signal Detection profile);

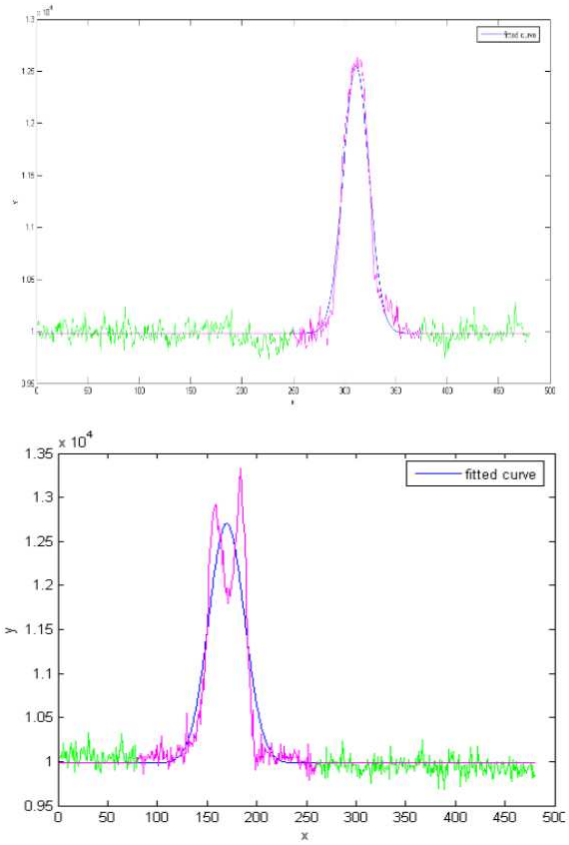

Figure 4: Example of region of interest selection for two different input signals; the green curve is the raw signal, blue curve is the fitted curve, and magenta curve is the output signal, after signal detection.

3. the baseline itself (*baseline*);

4. centroid (*AsdCentroid*) in pixel, area in A.U.(*AsdArea*), and rms width already converted to radians (*AsdSigmap*).

2.2 Signal Cleaning

The following step is trying to precisely discriminate where the beam signal starts, separating it from the sea of noise. This is probably the most important chain mail of all the procedure, since the output parameters will strongly depend on the choices done at this point. The core of the routine (*signalCleaning.m*) is an iterative procedure that, starting from the output signal of previous function, calculates the profile centroid and RMS width, then shrinks the region of interest to ± 3 rms around the centroid, and recalculates the same parameters again. Then it compares the new values with the old ones and keeps

going until they match each other. This procedure makes use of two different facts:

- after the detection and the baseline subtraction, the remaining noise in the signal vicinity fluctuates around zero, with positive and negative values; since the profile values represent a weight in the rms formula, negative numbers tend to decrease the rms value, so that until negative values are present in the signal, the window continues to shrink.

- In the signal vicinity the signal mean starts to increase from zero, since signal tails add to noise; moreover the baseline calculated with the fit inside the previous function is sometimes slightly underestimated ($0.1 - 0.2\%$), because of the superimposition of the Gaussian tails.

Together these two points make the algorithm work. At the first iteration the function calculates an rms value that leads to a smaller window than the actual one because of negative values, but bigger than the right one (that of an ideal profile with only signal, no noise and baseline), because positive and negative values are not perfectly balanced (second point in the above list); the second iteration does the same, but this time both the contributions (the one that tends to decrease and the one that tends to increase) will be smaller, because of less noise inside the window, and so on. At a certain point the beginning of the signal under the noise will lead to a rising mean and the two "forces" will come to a balance. Typically the procedure converges after $4-5$ iterations to the equilibrium point. The local mean of the signal acts as a friction, a "force against movement", where movement means the tendency to erode the signal, and its rising toward the signal makes the force strength directly proportional to the distance from equilibrium. Obviously a percent of the signal will be lost before reaching the balance. For a given level of noise, the absolute area of signal lost is more or less the same for every kind of signal, but the percent respect to the whole signal area depend on the signal-to-noise ratio (this point will be demonstrated with tests later in this section). The outputs are:

1. the cleaned profile, i.e. the portion of the profile that the program has recognized as signal(*CleanedProf*);

2. same as above but with baseline added (*CleanedProf_bsln*);

Lots of tests with different signal shapes and SNR have been done to validate this algorithm and will be summarized at the end of this section.

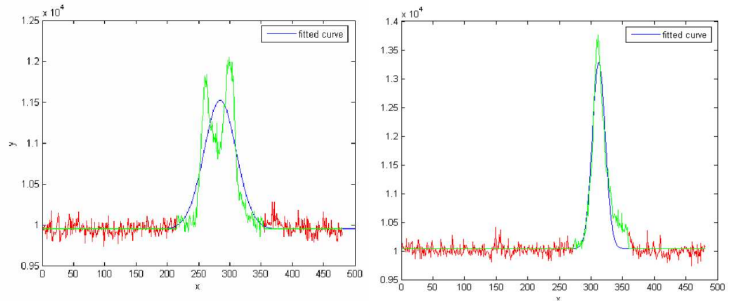

Figure 5: examples of different signals cleaning: red curve is the original signal, blue curve is the gauss fit to detect the signal, and green curve is the signal after cleaning. In the signal region the red curve is not visible because it is superimposed to the green one.

2.3 Results calculation and saving

The last step is to calculate the values for needed parameters (*SingleProfRmsCalc.m*). This is done using the same formulas as above(eqn.(2)(3)(4)), applied to the cleaned profile. The output of this step is a vector (*finalData*) with the values needed. After all, data are saved inside a folder (*"ElaboratedData"*) automatically generated by the program at the same level of the data folder, also reproducing the same data sub-folder tree. The output vector is then saved in the homonym elaboration folder of that from where images were loaded. The file that contains the elaborated data has the same name of the images folder (for example *fend01.txt*). Since this program has the possibility to reconstruct the phase space, one needs to save also the profiles (*prof01.txt*). Any of the above indicated profiles can be saved, but by default the raw profiles without baseline are chosen.

2.4 Tests on the image analysis routine

As said before, a big number of tests have been done on the algorithm to validate it. In the following we want to show the results of these tests: a set of profile vectors reproducing all the possible data profile has been created (with and without random noise), and given as input to the program; the comparison between the real profile values and those reconstructed by the program after the detection and cleaning procedures will be shown.

Looking at the data, one observes that all the possible beamlets distributions lead to 2 different type of profiles:

- signals symmetric respect to their maximum; this kind of data can be well represented as a truncated (or not) Gaussian;
- signal not symmetric respect to their maximum; in this case 2 Gaussian functions

are needed to represent it. The second Gaussian typically should look like a small bump on the main signal tail but, depending on the situation (for example in the solenoid scans), can be also at the same level or bigger than the first one. Tests varying the height and the distance between the two signals have been carried on. Obviously also the width of the second gaussian will vary, but the results are not reported, since the algorithm is not sensible to this parameter.

It is mandatory to start the algorithm test with a single Gaussian without noise, just to see what appends with an ideal profile. A Gaussian profile with different sigma and heights is used as program input, and results are summarized on the fig.(6).

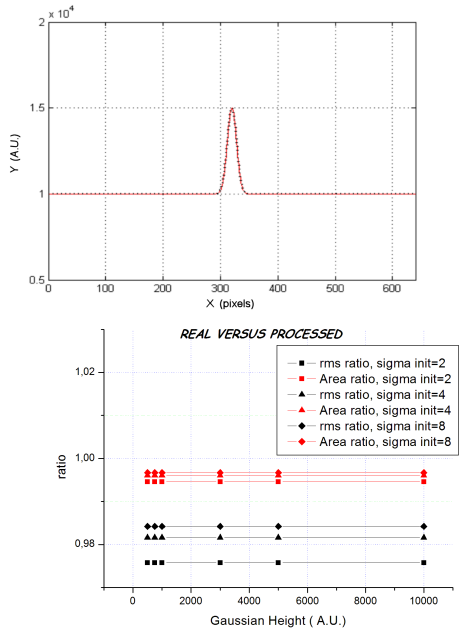

Figure 6: the upper figure shows a particular input test profile (height=$5000 A.U.$, $\sigma = 8 pix$) in red, superimposed to a black dotted curve, the processed profile. The lower graph summarizes the test results for different gaussian σ and heights. The numbers don't change with height because there's no change in SNR (the noise is zero).

The upper picture of fig.(6) shows a particular input test profile in red (height=$5000 A.U.$, $\sigma = 8 pix$, no noise added), superimposed to a black dotted curve, the processed profile. The lower graph summarizes the test results: on the x axis the height of the gaussian is reported, while the y axis represents the ratio of the areas (red markers) and rms widths (black markers) between starting (real) and processed profile. Different marker shapes

represent different σ of input gaussian, from 2 to 8 pixels. The reconstructed areas are almost independent from Gaussian parameters (the ratio tends to 1 increasing the rms of the initial signal, but it is always around 99.6%). In the case of rms widths the difference is slightly bigger, but always not relevant (not less than 97.5%) . The numbers don't change with height because there's no change in SNR (the noise is zero) In this case the difference between the real numbers and the reconstructed ones, is due to the fact that the program cuts the profile, identifying only the values inside ± 3 sigma around the centroid as signal. Obviously this choice is not the best for this ideal case (the rms calculated will be the sigma of 99% of signal), but it was found to be the optimum when the noise is added. After this point in fact the SNR become too small, and the rms calculated is affected overall from noise; moreover the value becomes very sensible to the last-point choice. The second test was done with a 3 sigma truncated Gaussian, adding a baseline but no noise, trying to see some difference from the above case. Results are summarized in fig.(7).

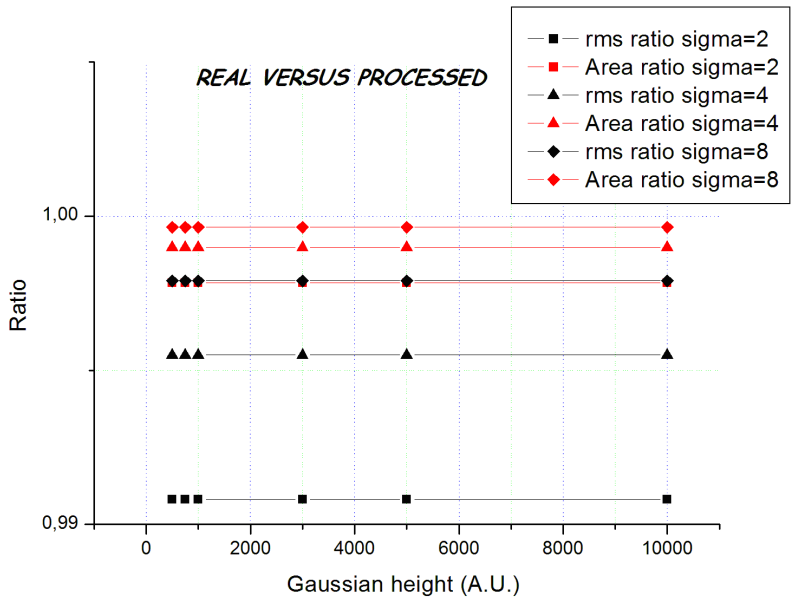

Figure 7: Results of the tests with a truncated Gaussian. In this case the ratio is very close to one.

This test has been done because the image profile to analyze won't have infinitely long tails but it'll drop off to zero (as every real signal does). Results are better than with the entire function both for area and rms width, but the latter profits more from the cut. It can be understood thinking that the same cut in area will produce different effects on the rms, depending on the distance between the signal mean and the cut-zone barycenter; in

our case tails carry a very small fraction of area, having however a non negligible weight in rms calculation.

It is worth to note that (as seen in the previous test with ideal gaussian without noise) the program works also for ideal profiles, although the mechanism that makes it work is different from that described before for the case of noisy signal: once the first region is selected from the fit, the program calculates the rms value and shrinks the window to 3 rms, wasting the 0.4% of the beam. At this point one has a truncated Gaussian, and the rms will be in the worst case 98% of the previous one (as can be seen from the fig.(6)). If the σ is such that the percent left is equal or more than 1/3 pixel ($3\sigma \geq 1$) the window will shrink again about that quantity, but this time the situation will be that of the fig.(7): in the worst case the new rms calculated will be 99% of the previous, and the algorithm will stop unless $\sigma < 34$ (1% of sigma should be bigger than 3, from the same condition as before). It is impossible in our case to have such a big value. This last example has been described just to understand the procedure, but in reality the limits are much less tight, since going up with the profile width, algorithm become more and more precise (see fig.(7)).

The next step is introduce noise and simulate a real input profile. First test were done with a single Gaussian, (truncated or not), adding a constant baseline (10000 A.U., the typical extracted value), and a random noise (white noise, uniform distribution) with a peak to peak values of 300. The SNR was varied changing the signal area and height. Moreover, for each profile a noise-dependency study has been done, generating 50 times the noise vector added always at the same signal and processing the overall profile, to understand the sensibility to spike position and to noise pattern in general. So from now on every graph will have points (mean values) with error bars, calculated as standard deviation of the mean divided for the square root of number of *"measurements"*(50). With the single Gaussian test we would like to demonstrate the limits of applicability of this algorithm, and also understand if there is a dependency of the results from position of beamlet in the image (the fit could work better in the central window zone than in the lateral). Figure(9) shows the results obtained from single gaussian tests. On the y axis is shown the ratio between real parameters (that calculated before adding baseline and noise) and the reconstructed ones, at the program exit. The first plot is the area ratio, that gives an idea on the fraction of the beam left out from reconstruction. Obviously it decrease with the increasing of SNR (defined from now on as ratio between Gaussian height and noise level ($300 A.U.$), so directly proportional to the x axis), because of the decreasing of the beam fraction hidden by the noise; the precision also raises with the signal width (different curves). The same can be argued for the second graph that represents sigma ratio between the real and the reconstructed rms. In this case the first points are less precise respect to the same in the upper graph, since the hidden signal is always on the tails and, although

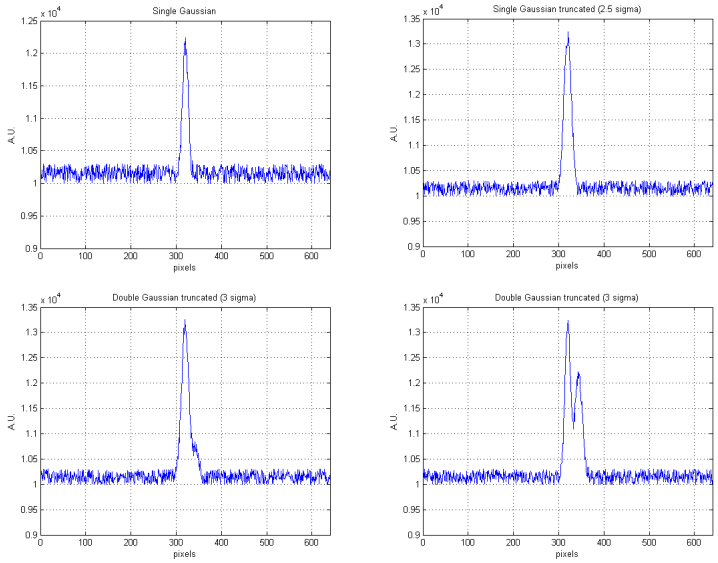

Figure 8: different examples of input profiles used as test; the upper left and upper right profiles are single Gaussian with different heights and sigma; moreover the second one is truncated to $\pm 2.5\sigma$. The lower right and left profiles make use of 2 Gaussian with different centers and widths. In both cases the entire signal is truncated to $\pm 3\sigma$. These last 2 profiles should reproduce real signals such as those of fig.(5).

the area fraction can be small, the tail contribution to the rms is relatively big. Also the error bars tend to decrease going from left to right, meaning that noise pattern sensitivity decreases. In both plots 90% precision is reached for the lower curve ($sigma = 5$), when the profile height reaches approximately 1500 A.U. (black vertical line).

Figure 10 shows results obtained from input signals with different centroids. Since the images are $640X480$ pixels, after the integration along one direction the resulting profile will have a length equal to one of those two values (depending on the direction of integration). In the case of fig.10 a test profile vector of 640 elements has been created, and the centroid profile moved first of all on the left ($xmean = 100$), then on the center ($xmean = 320$) and on the right ($xmean = 540$). The plot shows that there is not dependency from profile centroid position, since the algorithm outputs are the same for different signal centroid positions. The plot shows only test done with a Gaussian with $\sigma = 5$, being this the minimum width possible in our case (see 2 in 2) and therefore the most problematic.

Another point that comes out from the fig.(9) is that, also for big SNR, the reconstructed width never reaches the real value (then last point in the graph is 0.985). This is

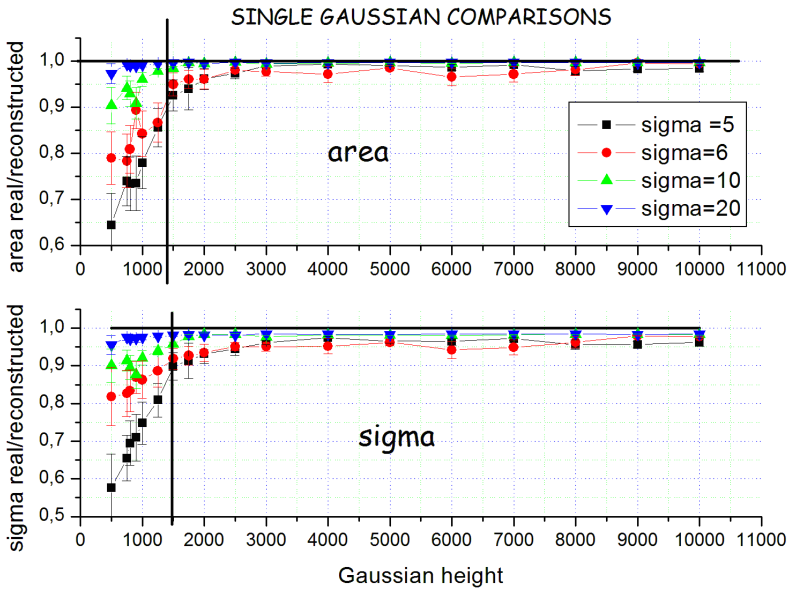

Figure 9: Results obtained from a noisy test profile as program input. Each curve has different width, so that it clearly shows that the precision of the reconstruction depends both on SNR or signal width. Dividing the x axis (Gaussian maximum) for the noise level ($300 A.U.$) one obtains the SNR.

a due, as said before, to the infinite tails of the test function used. Next graph (fig.(11)) shows the same tests as above done with a truncated (to 2.5σ) Gaussian, something closer to the real image profile. The main differences are the start values ($x = 500$) closer to the real one, and the "asymptotic value", equal to 1. The SNR value for which one has the 90% in the lower curve is always around 1500, but this is probably due to the fact that the difference between the two above cases tends to decrease as the profile intensity increases (the tails weight in the rms decreases).

Last test was done with a double Gaussian:

$$f(x) = a_p \cdot e^{\frac{(x-c_p)^2}{2\sigma_p^2}} + a_s \cdot e^{\frac{(x-c_s)^2}{2\sigma_s^2}} + h. \tag{6}$$

The first gaussian was left constant to $\sigma_p = 8pix$ and $a_p = 3000 A.U.$. the choice of these constant parameters has to be discussed: the double-Gaussian-like profile comes out when the beam is represented by an X-shaped trace space. In this case the profiles from central slits are unaffected, such as that from lateral-end slits, since the second branch is usually shorter and less intense. The problem is visible only in the slits between the center and the end, the middle-lefts and middle-rights, so that the principal signal has always an intensity and a width of the order of the fixed one. The parameters of the second Gaussian

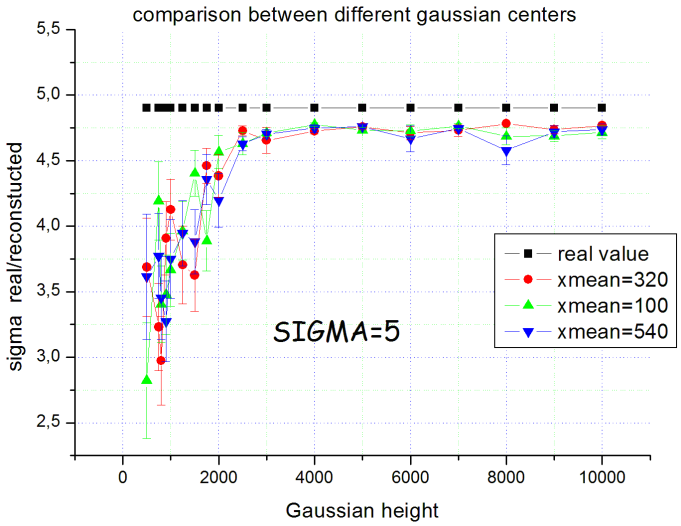

Figure 10: there's no dependency of final values from beamlet centroid position.

are varied according to the real data:

- Height a_s between 500 and 5000;
- Sigma σ_s between 2 and 8;
- Relative centroid distance $|c_p - c_s|$ between 1 and 3 times σ_p ;

In fig.(12) we summarize the test results for the maximum centroid relative distance ($3\sigma_p$), since this is the worst case for the analysis. The expected problem was the possible width underestimate for small values of second Gaussian maximum, since its area is small compared to the total signal, while its contribution to the second moment of the distribution is not negligible because the relatively big distance from distribution barycenter. The expectation was wrong, as can be seen from the graph of fig.(12), since also for small values of second peak, the algorithm succeeds to reconstruct the parameters, i.e. the ratio between the real and reconstructed parameters is always very close to 1. One probably needs to go even more down with values to see this kind of error. The error bars slightly decrease with SNR, being always below 2% and demonstrating a small dependency from noise pattern. On the next graph (fig.(13)) the more critical points are showed (those with minimum SNR) for different second-peak widths. This gives an idea on what is the maximum expected deviation from real in these cases, and its dependency from noise pattern.

Before the end of this chapter we want to give an idea on what are the two limit situations that can be encountered while analyzing data. We took two different sets of data, with different trace space shapes.

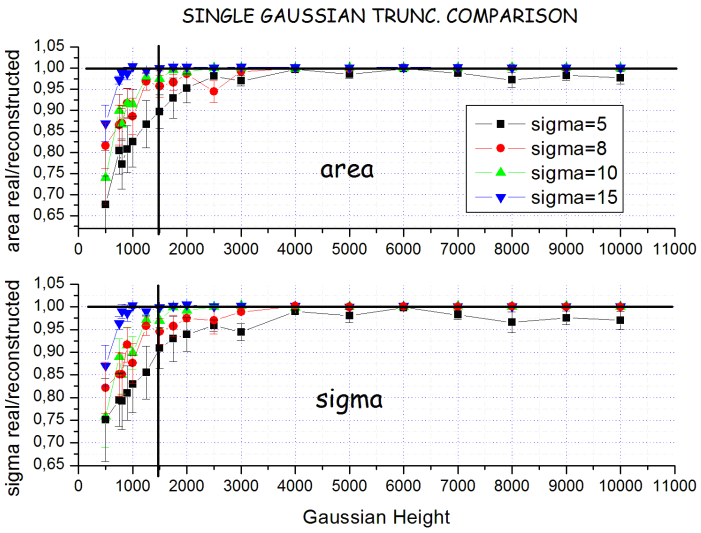

Figure 11: Test results with a truncated Gaussian; differences from the previous case are the start ratio (bigger in this case) and the asymptotic value, equal to 1.

Both the measurements were taken on November the 26-2006, in the same starting conditions (same solenoid current and RF phase), but at different locations. The first one was taken positioning the Emittance-meter at $1230mm$ from the cathode, while the second one was at $1589mm$ away from it. Figure(14) is a sketch of the profile rms width (the points have as error bar ± 3 times the profile rms width) as a function of the centroid position. It is not a trace space plot, since the orientation is wrong, but the thickness and the shape of the figure are directly proportional to those of the phase space plot (for its precise drawing we refer to the section (4)). As can be seen, the plots have different behaviors: in the first one the width increases moving away from the center, so that the profile with smaller area will have bigger rms; in the second sketch the opposite situation is observed, i.e. an inverse correlation between distance from centroid and width is found.

The physical explanation of the two different shape is out of the scope of this chapter, but qualitatively it can be understood knowing that first measurement (upper plot in fig.(14) was taken in the vicinity of the beam waist (after magnetic focalization by a solenoid lens) and its butterfly shape is due to different longitudinal slice waist positions. It is a direct consequence of beam correlated energy spread coupled with the lens chromatic aberrations. The second measurement (lower plot in fig.(14) was done at the beginning of the after-lens drift region, where the entire beam is convergent.

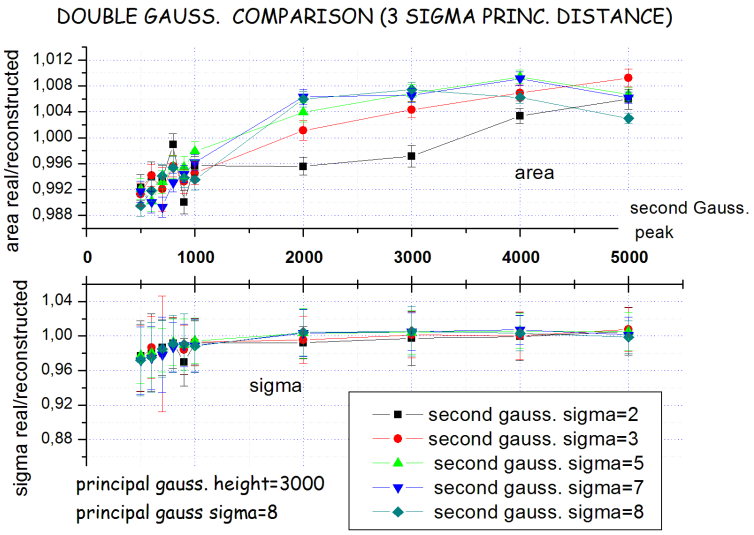

Figure 12: Results for the algorithm test with a double Gaussian. The SNR between the second Gaussian and the noise has been varied, leaving the first one fixed. The results are always in good agreement with real values .

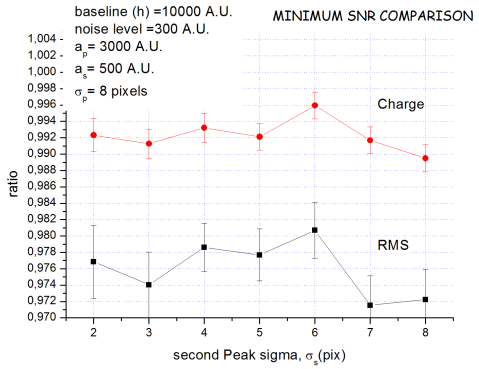

Figure 13: The graph represents a portion of the above points, those with the minimum SNR ($x = 500$).

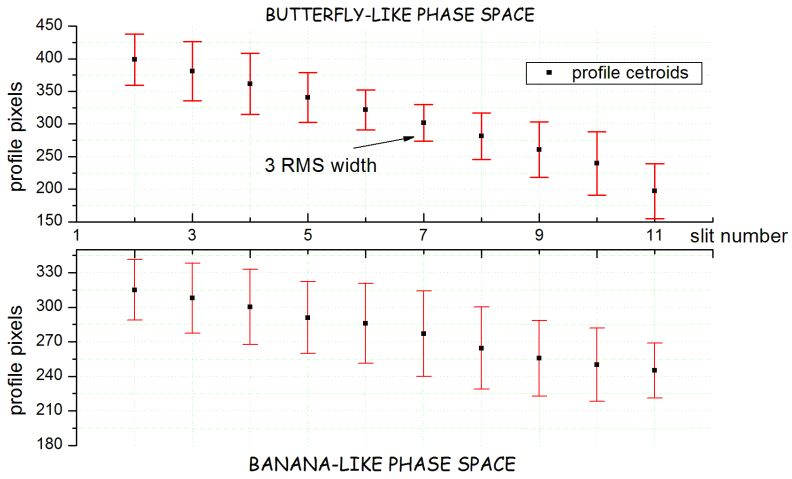

Figure 14: Two different beam behaviors, showing different space-divergence correlation. The upper plot shows a *"butterfly-like"* shape, while the lower one shows a *"banana-like"* shape.

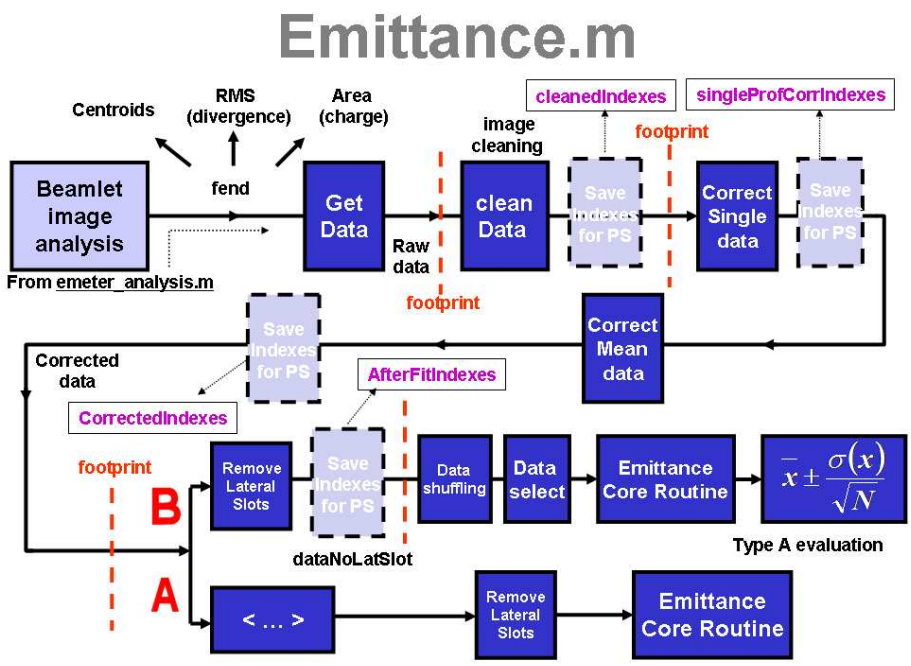

Figure 15: Dataflow of the *emittance.m* program.

3 Emittance extrapolation from data

Once the area, centroid and rms width have been extrapolated from each single image, covariance matrix of beam $(x - x^{'})$ distribution can be calculated. For each slit position in a measurement a certain number of images (variable from 10 to 30) is acquired, leading to the same number of values for each of the 3 parameters. All the reconstructed values for a single parameter should be identical for a fixed slit number and cathode distance, but in reality both the system instability (fluctuation of parameters from assigned value) and the (small) dependence of image analysis algorithm output from noise, lead to a shot to shot values fluctuations. Moreover electric field discharges in the gun, beam-RF dephasing and other temporary problems may lead to images without signal, or with signal properties not consistent with the specific situation.

Before to start the routine explanation, it is worth to underline that the control room measurement procedure is totally automatic, so that the operator has the only assignment to fix the starting position and push the start button.For instance a complete emittance scan for 30 different distances from cathode takes about 45 minutes; during this period everything can appends, from discharges to laser stop, while the acquisition continues, and only the operator problem-solving skill determines the quantity of blank or wrong data. With this in mind it becomes obvious that, analyzing data, a robust cleaning procedure

has to be implemented to reduce the effect of systematic errors.

The *Emittance.m* routine dataflow is showed in fig.(15).Taking as input the reconstructed values from *emeter_analysis.m*, the routine cleans the set of data from bad values, and calculates the beam parameters with to different methods, one making use of averaged slit parameters and the other that averages different emittances calculated using each single parameter. In latter case also the error bars can be calculated.

All subroutine called during the execution are represented by blue boxes. Following the timeline, the routine main tasks are:

1. Get data from files saved by the previous program, (*GetData.m*) ;

2. Single image cleaning; at this point any data vector of single image (area, centroid and RMS) is individually analyzed. The algorithm applies some thresholds, taking decisions of keep/discarding the specific image and the relative data vector(*CleanData.m*);

3. Single slit cleaning; every data vector relative to an image that overtakes the previous controls, is compared with the others belonging to the same slit position. Statistical quantities are used, such as mean and standard deviation of dataset. The tails of the distribution are trashed in the effort to reduce systematic errors and with the goal to leave only the most probable values, (*CorrectSingleData.m*);

4. Mean values cleaning; from the remaining values (dataset) a mean value per slit position is obtained, and looking to its standard deviations the quality of the specific dataset can be extrapolated, deciding whether it can be trusted or not, (*CorrectMeanData.m*);

From now on the routine dataflow bifurcates in two branches; the one below (branch A in fig.(15)) calculates the beam characteristics (ε, αandβ) from averaged parameters. From every slit position acquisition comes out only one vector of 3 mean parameters (calculated using values that survived to the controls described above), and only an emittance value is reconstructed from the following steps:

5a. A new average from the remaining values is calculated for every slit, and a mean vector (area, width and centroid) is produced, (*imageAveraging.m*);

6a. Mean values correction; mean parameters from different slits are now compared to verify the consistency of the whole set of data, and correcting strange and impossible behaviors (due for example to a system failure that extends for a period

comparable with a slit time measurement, 4-5 seconds). A linear fit of central slit centroids is carried on and deviations of mean values from linear behavior analyzed (*RemoveLateralSlots.m*);

7a. Emittance final calculation [2]; first and second order moments of both spatial and angle distribution are calculated, including their cross-correlation. Distributions are sampled by the slit positions, so that the moments are calculated as discrete sum of weighted values (the weight is represented by beamlet area), (*EmittanceCoreRoutine.m*).

The dataflow branch B an alternative way of data treatment. In this case the averaging is done at the end on the emittance value itself, i.e. the average is directly applied on the final result. Suppose for example to analyze a measurement with 15 images per slit position, and suppose that only 10 of them survive to cleaning and correction procedure; for every slit position one will have 10 areas, centroids and rms widths. Instead of averaging them and use their mean value to find the result, we randomly associate each single value with others relative to different positions, and calculate 10 different emittances. This can be done since now each single value measured is supposed to be independent from the others; eventual systematic effects that correlate data have been corrected at this point, and the remaining data fluctuations are the superimposition of different independent statistical fluctuation sources. With this method indeed all the possible sources of error are taken into account, from the instability of the whole apparatus to the reproducibility of the analysis software outputs. All together these sources build up in the error bars.

The procedure is as follows:

5b. The first subroutine (*RemoveLateralSlots.m*) has already been described. The only difference in this case is that while the fit is done on the mean centroids of central slits as before, the comparison is done between the fitted line and each single data, correcting in case the single data and not the mean value of the specific slit acquisition;

6b. Creation of datasets for emittance calculation; in the hypothesis that all data from successive shots are independent from each others, good data are reordered and randomly associated to others to create complete datasets for emittance calculations, (*ShuffleData.m*);

7b. Select the created datasets to use in following steps; in general each slit position will have different numbers of good data. In particular lateral positions often have

less usable measurements; if one does not want to use fictitious beam size has to be careful in this choice, (*DataSelect.m*);

8b. The same subroutine is used to calculate the emittance, (*EmittanceCoreRoutine.m*), but this time it is repeated for the number of independent datasets created;

9b. Arithmetic mean and standard deviation of emittance, alpha and beta parameters are calculated. Experimental standard deviation (uncertainty) is then found using the type A evaluation (i.e. dividing for the square root of number of data the statistical standard deviation, refer to ([1])).

In every kind of measurements there are 2 error sources: systematic and statistical errors . The task of the first four points described above, such as that of points $6a$ and $5b$, is the minimization of systematic errors, without affect statistical fluctuations. In the ideal case in which this goal is perfectly reached, the two methods should give close results, because statistical fluctuation does not affect the mean values. The differences in this case mostly depend on selection of dataset to use in the branch B. Moreover there is a theoretical difference: the mean of a function is equal to the function of the mean only for linear functions, and that is obviously not the case for the emittance as function of second order moments (see eq. (1) for rms emittance).

Before to go into details to justify choices done while trying to identify and exclude data affected by systematic error, a little explanation on the meaning of red dotted lines and dotted boxes in dataflow (Fig.(15)) is necessary. Red dotted lines represent diagnostic points in dataflow, where it is possible to save and plot information about excluded data until that point for each single folder, understanding what is the impact of every single subroutine on the total amount of data. Being 13 the number of slits positions, the totality of data for a single measurement is represented by a matrix with 13 columns and as much lines as the number of images collected for a single beam sampling. The subroutine (*footprint.m*) saves pictures for centroid, rms width, and area matrixes, representing as blue squares the excluded data, and as colored (128 colours) squares the good data (see Fig.(17)). These pictures, besides the information about the number of remaining images, give a qualitative information about the goodness of cleaning procedure: colours gradients can help the operator in identifying eventual non consistent data survived to controls. Moreover differences between before and after data shuffling can be understood and number of measurement to use in the branch B (*DataSelect*) chosen by eye.

The dotted boxes are points in which indexes of survived profiles can be saved in *.txt* file for future trace space reconstruction. Any of this point gives to the user the possibility of plotting (using another function later explained) the trace space, using only

profiles that reach a particular dataflow point (*FindBadProfiles.m*). A comparison between different trace spaces is possible in this way, helping to graphically understand which part of beam is trashed, in case it is, deciding for example if it is background noise or real signal. Violet words are the exact names of *.txt* files saved by the program. Dataflow objects representing these two last subroutine are drawn in dotted lines meaning that their execution can be excluded anytime from dataflow without affecting the final result.

Let's start with explanation of choices made to exclude systematic errors. First of all it has to be pointed out that the value of every threshold and meaningful parameter is asked at beginning by the program and can be changed anytime to study sensitivity to it. Here we want to argue the chosen default value.

We start from the Single signal cleaning routine (point 2 in the list). As said many times, from a single profile analysis 3 numbers are extracted: area, centroid and rms width. The considerations done in section (2) about the expected values, help us to fix some thresholds on these values:

- For centroids nothing can be said at this level. The center of the signal can be everywhere inside the image profile, so that looking at the single value there's no possible discrimination between wrong and right data;

- For the rms width we found a minimum of 5 and a maximum of 20 pixels; these numbers comes from the reasonable and experimentally verified hypothesis that the beam uncorrelated spread in angles is always more than 0.2 and less that $1mrad$. A threshold of $1.5mrad$ is therefore applied, meaning that profile width can not be more than

$$\frac{1.5 \cdot 10^{-3}(rad) \cdot 300 \cdot 10^{-3}(m)}{14.65 \cdot 10^{-6}(m/pix)} \cong 30pix; \tag{7}$$

  this constraint is not very stringent (10 pixels more than the maximum expected rms width), but at this level we want to exclude only the data that are for sure out of possible data range; the crossed control with centroid and area values will lead to the definitive choice. Situation of very big rms width can happens if the program fails to fit the profile (often because there is no signal), and tries to fit the background with a single Gaussian (in that case the rms will be a very big number);

- For the area a minimum value is fixed; respect to the previous case this happens when, failing to fit the signal, the program considers a noise peak as signal (fig(16)); sometimes the rms of the bump can fall inside the signal good rms range, but its area will always be less than the reconstructed from real signal. The value chosen as threshold is 2000 A.U.; it corresponds, considering the minimum signal width

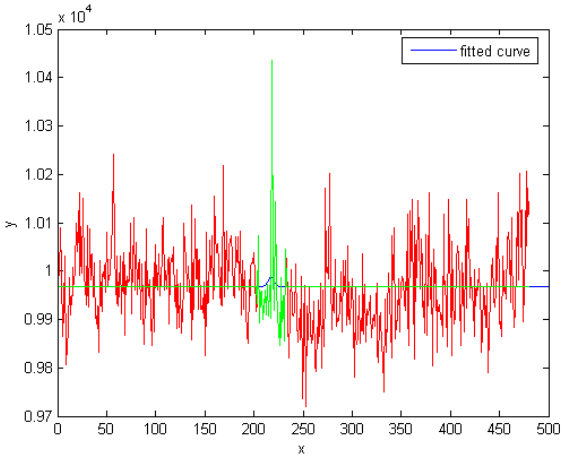

Figure 16: Example of noise peak fitting; red line is the raw profile, the small blue Gaussian is the program fit, and the green line is the ”signal” after the cleaning.

possible of 5 pixels, to approximately 200 A.U. as maximum signal height, a value below the expected noise level (300 A.U.).

The algorithm puts to -1000 (number used as tag, since nothing can be negative) all the values found to be beyond one of these thresholds, together with the other parameters relative to the same image. The area threshold seems too small, as the minimum signal area that one has is about 6000, but in this first phase the goal is only to exclude values that are completely out of range; later other controls, such as centroid consistency etc.., will be used to finely choose between remaining data.

Next step is the single slit position cleaning (list point n.3). This is a critical point since it is the last step before the calculation of mean values, the last chance to identify images with systematic errors before they affect the mean value leaving a that point the only solution to throw the entire slit measurement. What can be done is use statistical variables such as profile barycenter and rms width to isolate the points that have big distances from the rest of distribution. The best thing to do in this case is cut the tails of the distribution symmetrically, going always in the direction of *”most probable value”*; this procedure should cut at worst some good measures, leaving the mean value untouched (because statistical errors are symmetrically distributed, and cutting on both side doesn’t change the mean), while in best case it cuts only the wrong data. After some tests with the subroutine, a cut value of 1.5 times the distribution rms value has been chosen as the best threshold for our application. Every value out of ± 1.5 rms respect to the distribution centroid will be excluded at this point. The subroutine works at the separately on the centroid, area and rms distribution, synchronizing them at the end: it is sufficient one of

three values out of range to tag the specific measured image (profile) as wrong and force the three values to -1000. Test results are summarized in next points, for a measure with 15 images per slit:

- If all the images are good and the data distribution come from statistical fluctuations, normally from 1 to 4 images are trashed, depending on the amount of data statistical fluctuations. As said before this action doesn't change the mean, and it has the only final consequence to increase the error bars in the Branch B (the number of emittance values decreases);

- If only a fraction of data is incorrect ($1 - 5$), the job is done, since the wrong data are usually very far from right ones increasing the rms very much. On the other hand there's no big change in centroid position, leaving the real data close to it;

- If only a fraction of data is correct ($1 - 5$), it depends on how the incorrect data is placed around the correct set; if the distribution is symmetric and the center is close to good data, then the subroutine will succeed, otherwise a subset of data will pass this step, and the entire measurement will be thrown in the next control (on the mean parameters, list point n.4);

- If all the data are incorrect, then some of them probably will survive, especially if there is some kind of aggregation interval, where the density of points is higher. A mean value will come out from this dataset, but the consistency check with other slit values will exclude the value(list point n.4, or 6a/5b).

First two points are obviously the most frequent; the last one is generally individuate after a first data screening and the relative folder manually trashed; in the third case only a manual hint to the program can solve the situation.

The forth step of the algorithm is the correction of mean values. The data that passed previous controls are then averaged in each single slit position, and the standard deviation is also calculated. Before trying to compare different slits parameters to check the data consistency, some other control is needed. The standard deviation of the mean is an indicator of the dataset scattering. The transverse beam projection can reach at maximum $1cm$ total length, about $1.7mm$ rms width in both planes. Moreover, looking to the reconstructed trace planes, maximum total divergence measured is about $6mrad$. Knowing the number of beam samples, both the maximum distance between centroids in space and angles can be extrapolated. We usually cut the entire beam in 13 positions, so the difference in mean divergence between 2 consecutive slits, considering the limit

situation of a divergent beam with $6mrad$ spread in angles, is about $0.5mrad$ This lead to a max. centroid difference of:

$$\left(\frac{10000(\mu m)}{13} + 400(mm) \cdot 0.5(mrad)\right) \cdot \frac{1}{14.65(\mu m)} \cong 66pix \tag{8}$$

in case during the measurement the maximum possible distance ($400mm$) between the screen and the slit plane is used. Guessing a Gaussian distribution of points around the right value, and dividing that number 6 times ($\pm 3\sigma$), one gets a standard deviation of 11 pixels; on the other hand if points would be randomly distributed inside the window, the standard deviation should be $66/\sqrt{12} \approx 19$. The choice is on the middle of the two numbers, considering too strong the constraint of perfectly Gaussian distribution, but at the same time considering a uniformly distributed dataset to be noise (although it is only in a relatively small region of the window, a real signal with a randomly jumping centroid inside a 66 pixels region is to be considered as noise). The used threshold for centroid standard deviation is 15 pixels. Same thing is done on the divergence standard deviation, directly linked to the rms profile width. In this case the standard deviation used is equal to the maximum value possible, i.e. $1mrad$. Another control can be done on the minimum average area possible: with an rms value of 5 and an height of 500, in the worst case of a signal comparable with a single Gaussian truncated to ± 2.5 sigma, one finds (from the tests showed in section (2.4)) an area equal to approximately 6200. A lower limit value of 6000 is then chosen. This last control could be done also before, on the single image.

We prefer to do it at this point because a threshold on the average area seems to be more adequate, not affected by fluctuations. When applying some strong constraint on single image one run the risk to make the distribution no more symmetric, throwing only one side (that of smaller values) of it and changing the relative mean.

Last consistency control, is the centroid comparison, as briefly anticipated in point 6a and 5b. A check between calculated parameters in different positions is carried on. It is an important to exclude situations deriving from last two points in the above list; these pathological situations in fact can lead to a non negligible increase of emittance, if left uncorrected.

Nothing can be said *a priori* on the area distribution along the beam, since almost every charge-position correlation is possible during beam evolution; as seen before, also angles-position correlation can change from butterfly-like to banana-like correlation along the beam evolution. It is very difficult to extrapolate the difference between their measured distribution and the expected one, since there's no a real "expected distribution".

The situation changes for the centroid distribution; depending on the position of the slit plane respect to the beam waist, beam itself can be convergent, divergent or at waist.

In the ideal case of only linear forces acting on the beam and no trace space bifurcations, all the centroids should be on a line, whose angular coefficient depends on the particular above mentioned situation. The choice done here is to trust the central centroid positions, since the bigger signal height minimizes the analysis failure, and fit them, finding the "centroid line". Comparing the remaining centroids data (that of lateral positions) with the theoretical values from the fitted line, a vector of differences is produced. Now a maximum value for these differences has to be decided.

The deviation from linear behavior comes from both non linear dynamics or X shaped trace spaces. In the first case, charge, transverse and longitudinal density, energy, RF injection phase, solenoid current, play an important rule so that it is very difficult to theoretically predict what should be the maximum ratio between non linear and linear forces, situation complicated by the fact that any of the specified parameters has a big range of used values. It is easier to have a look to the trace spaces drawn and extrapolate the value from data. A value of 15 pixels maximum deviation has been found.

For the second case of X shaped trace spaces we can extrapolate a value from the Monte Carlo test showed in sec.(2.4)). The worst case is that in which one has a double Gaussian profile where the two functions have each the maximum rms and area possible and the maximum distance between them. So, referring to eqn. (6), taking the 2 function heights a_p and a_s as equal, the distance between them $|c_p - c_s|$ equal to $3\sigma_p$, and the 2 Gaussian rms widths $\sigma_p = 20pixels$ and $\sigma_s = 10pixels$, it comes out that the area of the principal gaussian (p) is approximately the double of the other (s). In this case the resulting total centroid (calculated as the weighted mean of individual centroids) has a distance from c_p, equal to $\frac{1}{3}$ of the distance between the two functions (it comes out form the area balance, i.e. if the 2 areas would be equal the resultant centroid would be in the center), i.e. 20 pixels:

$$\begin{aligned} c_s &= c_p + 3\sigma_p; \\ A_t = A_p + A_s &= \frac{3}{2}A_p = 2A_s; \\ c_{tot} = \frac{1}{A_t}(\sum_i A_{p_i}x_i + \sum_i A_{s_i}x_i) &= \frac{2}{3}c_p + \frac{c_s}{3} = \bullet \\ \bullet &= \frac{3c_p + 3\sigma_p}{3} = c_p + \sigma_p = c_p + 20. \end{aligned}$$

Being this number bigger than the other (15 pixels found before), we choose this value as threshold. Every centroid out of ideal line for more than 20 pixels is then forced back on the line, meaning that reconstructed value was wrong and that, most probably, we are in one of the two last situations described before. In this case neither the area and

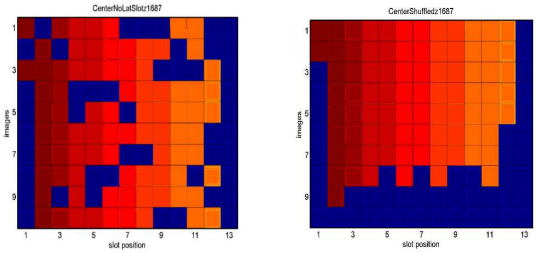

Figure 17: : data footprints; On the X axis the slit number, while on the Y axis the image number per slit position; as usual, the slit position along the beam are 13, while in this particular case 10 images per position were taken. Blue squares are thrown images, while the others are the remaining images; here the centroid footprint is reported; the slightly different colors indicate the different centroid values, that in this case raise from right to left. The right picture indicates same dataset of the left picture, but after the shuffling procedure.

the rms can be trusted, so that their value is forced to 0.

Now data are ready to be used to calculate the beam moments. From dataflow branch A emittance, alpha and beta parameters are reconstructed from mean values, using the formula derived in [2], that make use of only these 3 measurable quantities, plus other mechanical dimensions, such as slit width or slit plane-screen distance etc, for the calculus. To let the branch B reach the end another step has to be explained: the Data selection. At the end of the control procedure, different position have different numbers of survived images; while this is not a problem for the branch A algorithm that calculates the mean and doesn't care about the population, it is crucial for the other one, that uses any single value to calculate the emittance.

To better understand in fig(17) we reported the output of the *footprint.m* subroutine.

The right picture shows the same dataset of the left one, but after the shuffling. All the data have been reordered changing the relative lines by pushing as much as possible the squares towards the top of the matrix, such as the first line has the maximum number of positions with data. From each of these lines an emittance value can be calculated, but a decision has to be taken whether to use or not a line. Different lines in fact have different numbers of good data, implying fluctuations in beam transverse dimensions. But are they real or just consequences of the smaller percent of algorithm success with smaller SNR? A precise answer to this question is not possible, but the choice should be done looking at the shuffled footprint: until the good squares are all consecutive, data should be considered as good (in the particular case showed in fig(17), until line number 7, i.e. the first 7 images, leading to 7 emittance values) , because a beam transverse dimension fluctuation is possible and has to be considered. Moreover if the beam wouldn't fluctuate,

there is no reason to think that the algorithm would behave differently. The choice is taken once chosen the value of the variable "level", asked at the program begin, and having 1 as default value. This variable can go from 0 to 4 and it works as follows: once data is shuffled, number of good data per line is calculated. If level=0 then only the lines with the maximum number of images is taken into account to emittance calculation (the first 2 lines in the above case); if level=1 also the next group of lines, that with the higher number of good images but the previous ones (3rd, 4th and 5th lines), are used to calculate an emittance value (so emittance values from the first 5 lines will be calculated), and so on for level$= 2, 3, 4$.
The *footprint.m* subroutine makes use of a very useful way to represent data, because an eventual wrong value is immediately shown by its different color.

After data has been selected, next subroutine calculates the emittance in the same way of the other branch, but this time it is repeated as much times as number of lines chosen by data selection. Only after this step, the mean emittance, alpha and beta values are calculated. Error bars come from the type A statistical evaluation definition of measurement. The error bars represent the uncertainties derived by a statistical treatment of the data. First the mean and standard deviation of the quantity q (the emittance, alpha or beta in our case) is calculated:

$$\bar{q} = \frac{1}{N} \sum_{K=1}^{N} q_k; \tag{9}$$

$$s\left(q_k\right) = \sqrt{\frac{1}{N-1} \sum_{K=1}^{N} \left(q_k - \bar{q}\right)^2}; \tag{10}$$

where N is the number of *independent* measurements. The standard deviation calculated, doesn't depend on the number of measurements, since having 10 or 1000 measurements distributed on the same curve doesn't change its standard deviation. On the other hand is intuitive to understand that more measurements lead to more confidence on the mean value calculated. This is contained inside the mentioned *Type A* uncertainty definition, that decreases as number of measurements increases:

$$u\left(q\right) = s\left(\bar{q}\right) = \frac{s\left(q_k\right)}{\sqrt{N}}; \tag{11}$$

It represents the uncertainty on the measured mean value after N independent measurements, and will be the length of the error bars on emittance graph.

If one wants to know how big is the interval in which measurand expected values can fall, first of all it has to fix the fraction of probability values distribution to encompass ($68, 90, 95, 99\%$, etc..). Then using as value distribution the Student's distribution with a

Table G.2 – Value of $t_p(\nu)$ from the t-distribution for degrees of freedom ν that defines an interval $-t_p(\nu)$ to $+t_p(\nu)$ that encompasses the fraction p of the distribution

Degrees of freedom	Fraction p in percent					
ν	68,27[a]	90	95	95,45[a]	99	99,73[a]
1	1,84	6,31	12,71	13,97	63,66	235,80
2	1,32	2,92	4,30	4,53	9,92	19,21
3	1,20	2,35	3,18	3,31	5,84	9,22
4	1,14	2,13	2,78	2,87	4,60	6,62
5	1,11	2,02	2,57	2,65	4,03	5,51
6	1,09	1,94	2,45	2,52	3,71	4,90
7	1,08	1,89	2,36	2,43	3,50	4,53
8	1,07	1,86	2,31	2,37	3,36	4,28
9	1,06	1,83	2,26	2,32	3,25	4,09
10	1,05	1,81	2,23	2,28	3,17	3,96
11	1,05	1,80	2,20	2,25	3,11	3,85
12	1,04	1,78	2,18	2,23	3,05	3,76
13	1,04	1,77	2,16	2,21	3,01	3,69
14	1,04	1,76	2,14	2,20	2,98	3,64
15	1,03	1,75	2,13	2,18	2,95	3,59
16	1,03	1,75	2,12	2,17	2,92	3,54
17	1,03	1,74	2,11	2,16	2,90	3,51
18	1,03	1,73	2,10	2,15	2,88	3,48
19	1,03	1,73	2,09	2,14	2,86	3,45
20	1,03	1,72	2,09	2,13	2,85	3,42
25	1,02	1,71	2,06	2,11	2,79	3,33
30	1,02	1,70	2,04	2,09	2,75	3,27
35	1,01	1,70	2,03	2,07	2,72	3,23
40	1,01	1,68	2,02	2,06	2,70	3,20
45	1,01	1,68	2,01	2,06	2,69	3,18
50	1,01	1,68	2,01	2,05	2,68	3,16
100	1,005	1,660	1,984	2,025	2,626	3,077
∞	1,000	1,645	1,960	2,000	2,576	3,000

[a] For a quantity z described by a normal distribution with expectation μ_z and standard deviation σ, the interval $\mu_z \pm k\sigma$ encompasses p = 68,27, 95,45, and 99,73 percent of the distribution for k = 1, 2, and 3, respectively.

Figure 18: table of expanded uncertainty. The number found from this table once decided the fraction of distribution of values to encompass and calculated the number of degrees of freedom, has to be multiplied for the uncertainty to find the absolute value.

number of degree of freedom equal to the number of independent measurements (calculated emittance values), the expanded uncertainty can be calculated from the table shown if Fig.(18) ([1]). The number found has to be multiplied for the uncertainty:

$$U_p = k_p \cdot u\,(y) \rightarrow y \pm U_p; \tag{12}$$

Where K_p is the value found in the table, $u(y)$ is the measurand uncertainty, y is the measured value, and U_p is the resulting expanded uncertainty.

The program outputs are, besides already mentioned indexes files for future trace space reconstruction and footprint images, the *EmitData.txt* file, where the emittance, alpha and beta values from both procedures are saved together with the emittance uncertainty and the reconstructed entire beam rms.

4 Plotting the Trace space

4.1 Trace space reconstruction

More than first and second order moments can be extrapolated form the slit measurement method. In particular, the entire distribution in space and angles can be reconstructed from the sampling. Obviously the smaller spatial and angle distribution variation one can reconstruct depends on the sampling rate. It is not critical for us since one is interested above all on the trace space distribution envelope. We typically make use of 13 position along 6 beam sigma (3 around the barycenter), one sample every $\sigma/2$;

Once the trace space is reconstructed one can basically calculate any order of moments, not only the first and the second, having a complete and quantitative feeling on what is the beam quality in that plane. In the *emeter_analysis.m* and *Emittance.m* routines we already put the basis for the trace space drawing, choosing the type of profile to save and the level of correction to use for profile exclusion by selecting the indexes to save (*FindBadProfiles.m*). From the first choice depends the profile portion used for the reconstruction: one can use raw profiles with or without baseline (this second option is better because it increase the final plot dynamics), profiles after signal detection (*ASD_prof*), in which all out of 5 sigma around the centroid is thrown, or cleaned profiles (*cleanedProf*) with only the final profile portion used for parameter calculation remaining.

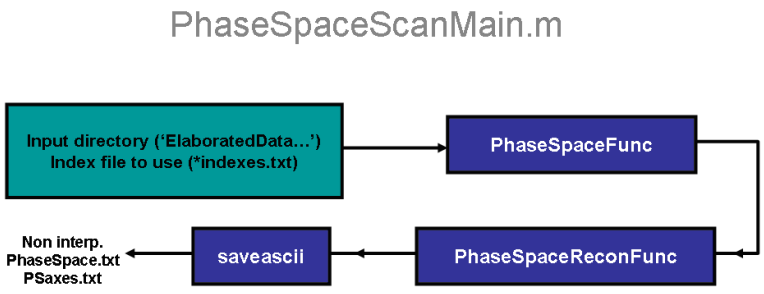

Figure 19: Trace space reconstruction dataflow.

From the second choice depends the number of profiles used to draw the plot: as explained in the previous chapter, indexes for profile exclusion can be saved after any routine step (single image cleaning, single slit cleaning, mean correction and centroid fit) and now used to exclude or not images that don't pass a particular control.

The main program is *PhaseSpaceScanMain.m*; it first asks for the scan directory and for the type of data to use (profiles and indexes). Then it calls the subroutine *PhaseSpaceFunc.m* that works into the single measurement folder inside the scan directory;

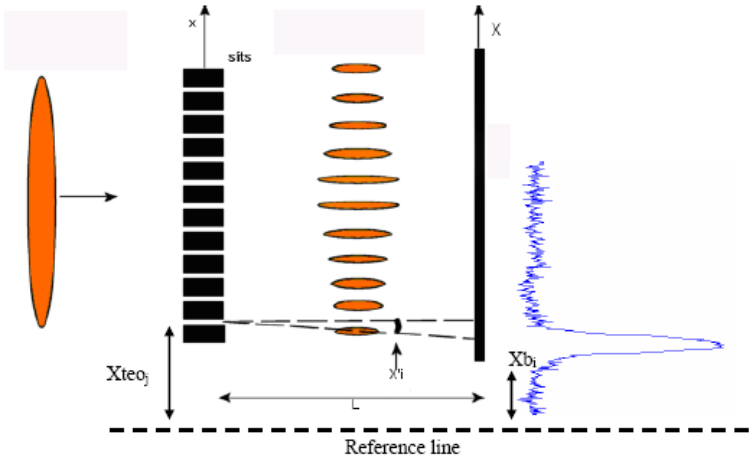

Figure 20: example of divergence-profile pixel association. In this case the reference line is the same for the slit centroid and for the profile pixels; they can also be different, since the only quantity with physical meaning is the relative difference between different positions. Once the two reference lines are chosen, they have to remain the same until the end of the measurement.

it gets the data from already saved files assigning them to variables, and scan the *info.txt* file to read the specific calibration constants (distance between different slit positions and distance between slit plane and imaging screen). All the selected profiles taken are then averaged for each slit position, resulting at the end in one matrix having 13 columns filled with mean profiles, and as much lines as profile length (480 or 640 depending on the plane considered). Then it calls the sub-subroutine *PhaseSpaceReconFunc.m*, the core of reconstruction procedure.

For every position one can assign to each profile pixel a specific divergence value. The first thing to do is fix one reference line for the slit position and another for the image pixels. This choice is completely arbitrary, and it will affect only the final trace space center, but neither its shape nor its moments. In fig.(20) the same reference is chosen, but only to simplify the picture. In the algorithm the slit position values are centered respect to the mean position value, i.e. a mean value is calculated and then subtracted to each single position. This is possible since the number of steps between each position is decided and saved by the control software after acquisition, and these numbers are readable from file (*info.txt*) in any moment without the need to analyze the data. The same can not be said for the images; the centroid of the entire beamlet distribution will be extrapolated only after the dataset has been analyzed; since we want a phase space plot which is independent from the emittance analysis (to study the convergence of different methods, and using one to understand the limits of the other), we don't want to use the

other procedure results. So the starting beamlet reference line will be the pixel 0 (nothing will be subtracted). Starting from the mean profile matrix, another matrix of the same size is created, whose elements will be the divergence associate to the relative element of the profile matrix. For the j-th profile (which means the j-th matrix column or the j-th slit position), the relative divergence vector will be:

$$[x_i^{'}] = \frac{([X_i]_j \cdot pixCal) - Xteo_j}{L}; \tag{13}$$

where *pixCal* is the pixel calibration, i.e. the device imaging system magnification, L is the distance between slit plane and imaging screen, and $Xteo_j$ is the projection of the j-th slit center on the imaging screen (a beamlet with zero mean divergence should have the centroid coincident with it). At this point the new matrix elements have the dimensions of radians; maximum and minimum matrix values (*minDiv* and *maxDiv*) are saved at this point for future applications. Now to make a trace space plot one needs to convert these numbers in indexes; in other words, we already know the X axis values, that are given by the slit positions, but we need the profile positions along the Y axis. The matrix of divergences has to become the matrix of ordinate indexes; to do so one has to divide the divergence matrix elements for the minimum visible divergence step, given by the image calibration and by the screen-slit distance:

$$x_{\min}^{'} = \frac{pixCal}{L} \tag{14}$$

The new matrix will have only integer numbers, but the arbitrary choice of reference lines can lead to negative numbers, so that it is necessary to add to it its minimum value ($+1$), making the elements become positive numbers, able to be real matrix indexes.

A new matrix is now created, having a number of lines equal to the max element value of indexes matrix, and 5 more columns, to leave blank space from the plot and signal begin and end. This new matrix is initialized to zero, and then filled with profiles in the respective columns, each profile element having as line index the respective value in the indexes matrix. Last but not least, a vector of real coordinates values has to be created. For the X axis, at this level the distance between consecutive pixels will be equal to the relative slit position distances, while the zero will be the calculated as the center of positions. For the Y axis, the pixel width will be equal to the $x^{'min}$ calculated before. The entire divergence interval is then calculated taking the difference between the maximum and the minimum values in divergence matrix (*minDiv* and *maxDiv*); this two number aren't symmetric respect to zero, their absolute value depending on the reference line. The arbitrary choice done here is to make the interval symmetric, i.e. make equals the *minDiv* and *maxDiv* absolute values. Both the phase space matrix and coordinates

vector are then saved in the *ElaboratedData* folder in two different files (*PhaseSpace.txt*, *PSaxes.txt*).

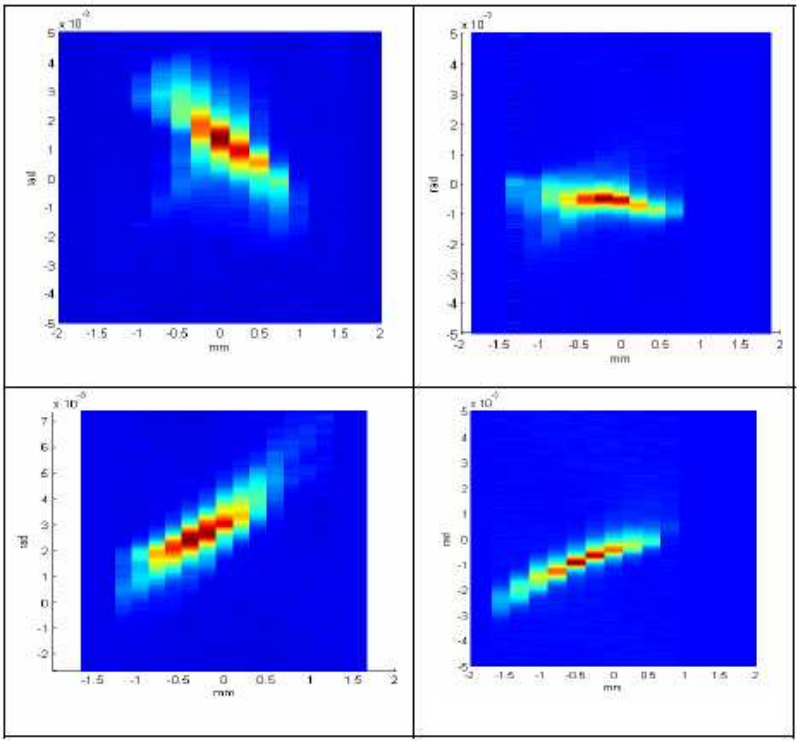

Figure 21: Reconstructed trace spaces in different positions; from the up-left: convergent beam, beam at waist, divergent beam, divergent beam. The images are reconstructed also with different profile types, so that different levels of noise are visible around the beam. A difference in image centroid is clearly visible.

4.2 Trace space interpolation

The reconstructed trace space has a good resolution in Y axis, but a poor X resolution that leads to images hardly usable for "*by eye speculation*". Because of this difference between the two planes (see fig.(21)), a two dimensional interpolation is not suitable to improve image appearance since it increases the both the number of lines and columns, adding useless data. Moreover the interpolation is always carried on along straight lines and columns; in our case the output image will be distorted since there is a correlation between planes. while in this case the interpolation should be done along the trace space axis (thinking to the signal as an ellipse);

The built-in MATLAB interpolation function creates fictitious undulations and an unreal signal lengthening with isolated islands of signal (fig.(23)). It derives from the

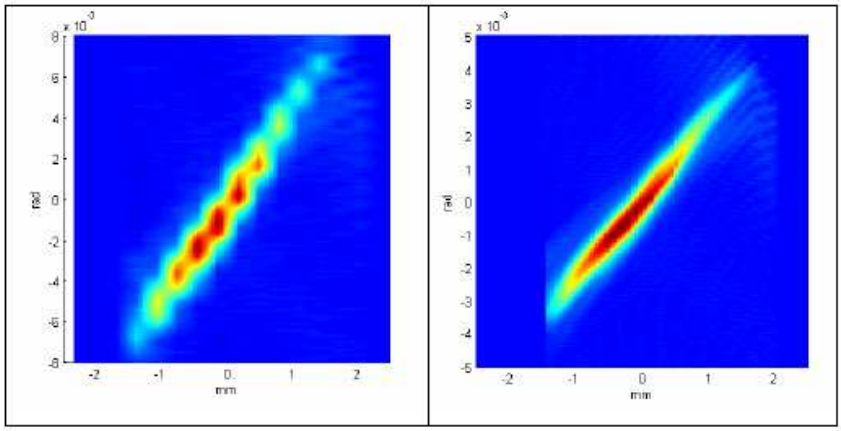

Figure 22: Difference from 2D MATLAB linear interpolation (left) and the manual " home made" interpolation (right) used in the program.

information about signal correlation, not used in such procedure. A manual interpolation procedure is implemented tacking into account this correlation: it is a $1D$ interpolation along X axis (since we need to improve signal quality only along that axis), applied on profiles aligned about their maximum value. Some small particulars has to be adjusted, as that of controlling the interpolation between one profile with signal and the another with only noise (typically this happens in lateral slits), that crates unreal signal wakes since the maximum value of a noisy profile can be everywhere. This is implemented applying another threshold: the Y distance between two maximum values of the two central profiles is calculated ($GapComp$), and every other distance (Gap_i) is compared with this number. If a specific difference is bigger than 4 times the measured value, than the interpolation between the relative profiles is done aligning them with the same gap of the central slits ($Gap_i = GapComp$), i.e. forcing back the profile maximum on the ellipse axis.The choice multiply 4 times the found value comes from experimental evidences, i.e. the tails often deviates from the central centroid line, and this choice as been found to be a good compromise, since also wrong profiles that could fall inside this range don't affect too much the interpolation, and the consequent analysis. Interpolated coordinates are obtained from the previous one just inserting between 2 values a third one equal to their mean. The interpolation is repeated 3 times (this number can be changed very easily only changing the $numIter$ value) and results, starting from the non interpolated trace spaces images showed before, are presented in fig.(23). This time the whole image is centered respect to the maximum value making invisible every centroid fluctuation.

From these trace phases, a lot of beam characteristics can be extrapolated, and also beam parameters such as emittance, alpha and beta, can be calculated, comparing results

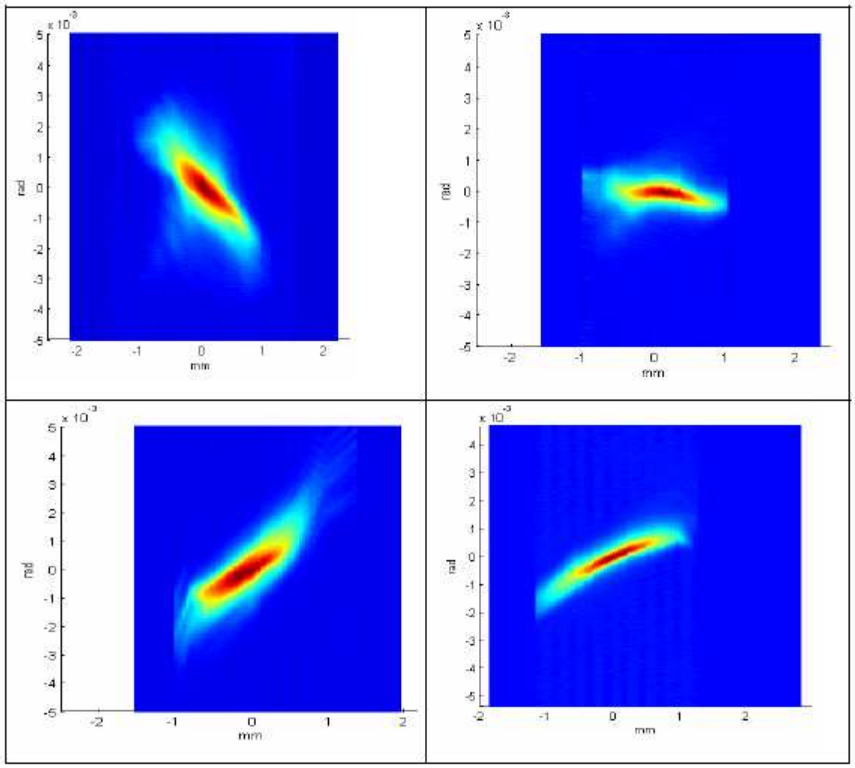

Figure 23: same images of fig.(21) after the interpolation.

with the others obtained from the independent way described in section (2) and (3). Refer to [3].

Acknowledgments

The author wants to thank all the people that gave him an hint during period needed to create, write and debug the program. All the SPARC data analysis group, Andrea Mostacci, Massimo Ferrario, Alessandro Cianchi, Enrica Chiadroni, Concetta Ronsivalle, Mauro Migliorati, Manuela Boscolo, Michele Castellano, that helped me in debugging the software, and from which i received advices that improved the robustness of the program.

A special thanks goes to my tutor, Prof. Palumbo who gave me the possibility to work in such exciting working group.

The bigger improvements to my work came always from the informal conversations with the brand-new prof. Pietro Musumeci.

The analysis software would have no meaning without the data. So a final thanks is addressed to the entire SPARC team.

References

[1] R.J. Barlow, *Statistics: a guide to the use of statistical methods in the physical sciences*, (John Wiley & Sons,Chichester, England, 1995).

[2] Min Zhang, *Emittance Formula for slits and pepper-pot Measurements*, FERMILAB-TM-1988, (Oct. 1996).

[3] P. Musumeci, D. Filippetto, *Trace Space reconstruction data analysis*, SPARC technote, 07/001, (Jan. 2007).

[4] C. Limborg, S. German, J. Power, *A modified quadscan technique for emittance measurement of space charge dominated beams*, Proceedings of PAC, (2003).

[5] J. Buon, *Beam phase space and emittance*, Proceedings, General accelerator physics, vol. 1 89-115, CERN, Geneva (1992).

[6] S.G. Anderson,J.B. Rosenzweig, G.P. LeSage, J.K. Crane, *Space Charge effects in high brigthness electron beam emittance measurements*, PRST. AB, vol.5, 0014201 (2002).