Labrujère's Problem

In Februari 1976, Dr. Th.E. Labrujère, at the National Aerospace Laboratory NLR, the Netherlands, wrote a Memorandum [1] which is titled, when translated in English: *The Least Squares Finite Element Method [L.S.FEM] applied to 2-D Incompressible Flow around a Circular Cylinder.* To be more precise: incompressible *and irrotational* flow. In this report, it was firmly established that a straightforward application of the Least Squares Method, using linear triangular Finite Elements, quite unexpectedly, *does not work well* ! Herewith, Labrujère's memorandum is demonstrating a scientific integrity which is rarely seen these days. With our own software the negative result obtained by NLR may be reproduced, exactly as it is.
Improving on these results has been a non-trivial task. On the side of NLR, it could only be accomplished by introducing highly complicated elements. On the side of myself, it could only be accomplished by adopting an approach which is quite deviant from the common Finite Element methodology. It has to be decided by Occam's Razor which of the two approaches is to be preferred.

The Calgary Solution

In December 1976, Labrujère's problem was "solved" by G. de Vries, T.E. Labrujère himself and D.H. Norrie, at the mechanical Engineering Department of The University of Calgary, Alberta, Canada. The result is written down in their Report no.86: A Least Squares Finite Element Solution for Potential Flow. The abstract of this report [2] is quoted here without permission:

The least-squares finite element method is formulated for the two-dimensional, irrotational flow of an incompressible, inviscid fluid. The continuity requirements on the components of velocity are established and shown to be more stringent than previously accepted. The development of the solution procedure is therefore based on fifth-order trial functions for both components of velocity. Using the least-squares procedure, the flow past a circular cylinder is calculated and the results shown to be in very close agreement with the theoretical solution.

End of quotation. Start of private opinion. It seems to me that the above solution is of pure academical interest, though. The apparent need for fifth-order trial functions will make this method *completely unworkable* in practice. Even if attention is restricted to the simple case at hand, I think it is way too complicated. What's worse, generalization is likely to be hard. In the end, 2-D and 3-D Navier Stokes equations (at a curvilinear grid, preferably) need to be solved. So the point of departure must be something which is much easier. Especially the number of unknowns at each nodal point should not exeed the absolute minimum, the number of degrees of freedom: two. I have never been in doubt that an alternative least squares finite element solution, *having* such desirable properties, must be possible. End of private opinion.

Incompressible irrotational (ideal) flow of an inviscid fluid is described by the following system of linear first-order (!) Partial Differential Equations (PDE's):

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \qquad : \text{incompressible}$$

$$\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} = 0 \qquad : \text{irrotational}$$

Here: (x, y) = coordinates , (u, v) = velocity-components.
There does *not*

It is often advantageous to carry out a Numerical Integration, instead of an "exact" one: see Zienkiewicz [3] chapter 8.8. This means that function values are to be determined at so-called integration points p. With each integration point p a certain weight factor w_p is associated:

$$\sum_E \sum_p w_p \left\{ \left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right]_p^2 + \left[\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right]_p^2 \right\} J_p = \text{minimum}$$

Here J_p is the Jacobian (determinant) of a transformation from global to local coordinates (see my articles elswhere on 'Triangle / Quadrilateral Algebra'). The jacobians J_p as well as the weighting factors w_p are positive real-valued numbers.

What follows now is a small step for man: *unify* the summations over the elements and the integration points, resulting in one global summation over all integration points ($i = E, p$), where (i) becomes the global index of any "integration point". This merely says that summing over elements, together with their integration points, is equivalent with summing over all the integration points in the whole domain of interest, in one big sweep. In this way, integration points can be interpreted as more elementary than the elements themselves. And an element with more than one integration point can be considered as a superposition of elementary integrated elements, with only one integration point (i) in each of them:

$$\sum_i w_i \left\{ \left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right]_i^2 + \left[\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right]_i^2 \right\} J_i = \text{minimum} = 0$$

In order for L.S.FEM to work properly, the minimum required must be a small number, rapidly approximating zero, as the size of the elements becomes smaller. Thus maybe it would be not such a weird idea to demand that this minimum value should merely *be zero from the start.* But then the above "variational integral" would have been equivalent to an non-squared system of equations. Because *when* a sum of squares can possibly be zero ? If and only if each of the separate terms in the sum is equal to zero:

$$\left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right]_i = 0 \qquad \left[\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right]_i = 0 \qquad : \text{for each integration point (i)}$$

Let's go one more step further. It is realized that each 'integration point' in the grid does in fact nothing else than contributing two independent equations. All integration points together contribute to the fact that a whole system of equations emerges in this way. Nothing prevents us from calling this a "Finite Difference" system of equations. Let's therefore, at last, replace the notion of 'an integration point' simply by: 'an F.D. equation'. And here we are !

Conjecture. *Any feasible Least Squares Finite Element Method is equivalent with forcing to zero the sum of squares of all equations emerging from some Finite Difference Method. Therefore L.S.FEM gives rise to the same solution as an equivalent system of finite difference equations.*

We are ready now to look at Labrujère's problem in the following way. Let it be required that the Least Squares Finite Element Method always leads to an acceptable solution, with moderate mesh sizes. Then, of course, in the associated Finite Difference system, the number of unknowns N should be *equal* to the number of independent equations M. If such is not the case, namely, then the system is likely to be overdetermined. And it is doubtful if the Least Squares minimum can still approach zero, fast enough. A simple count of the triangles involved with Labrujère's problem reveals that such kind of a delicate balance between unknowns and equations is definitely *not* achieved there: the number of elements outweights the number of nodal points by a factor 2 ! This means that there are roughly twice as many "unsquared" F.D. equations as there are unknowns. Apart from of any more complicated kind of argument, like higher order continuity, this surely throws up a basic question.

I am not qualified to check out whether Norrie and DeVries implicitly adressed that question, in [2]. They first kept the triangular shapes. I guess that, in order to compensate for an abundance of elementary equations, they had to introduce even so *many* additional variables. Now it becomes clear what kind of different approach may be feasible here. For the *only* thing that has to be accomplished is: a perfect balancing between the number of equations and the number of unknowns. Instead of increasing both these numbers again and again, why not better KISS: Keep It as Simple and Straightforward as possible.

Linear Quadrilaterals

So far, so good. But finding the right element really is a *critical* issue here. Preferrably something which is as simple and straightforward as the common Linear Triangle. Does it exist ? It certainly does ! Reference is made to a web publication called 'Quadrilateral Algebra'. It can be found as one of the PDF references, associated with the following place on the www :

`http://huizen.dto.tudelft.nl/deBruijn/programs/delphi.htm#NLR`

So, besides the linear triangle, there also exists a *linear quadrilateral.* The nodal points in this quadrilateral element are to be located in the middle of the edges; they are *not* coincident with the vertices. It is demonstrated in the reference that the element is accompanied, too, with additional equations for the unknowns, one for each degree of freedom. These additional equations do not impose any restriction on generality, because they stem from the fact that the midpoints of the edges in a quadrilateral always form a paralellogram. And due to isoparametrics, also the unknowns at the nodes must, so to speak, "form a paralellogram", in their own parameter space. The numbering conventions for the vertices and the midside nodes of the quadrilateral element are shown here:

```
4 _____3_____ 3
 |           |
 |           |
4             2
 |           |
 |_____ _____|
1      1      2
```

The quadrilateral, as defined by its vertices, will be called the *parent* element of the quadrilateral with its corners at the midside nodes.
Identities for the coordinates, when looking at the vertices, trivially are:

$$\begin{aligned}
\frac{1}{2}(x_1+x_4)+\frac{1}{2}(x_2+x_3) &= \frac{1}{2}(x_1+x_2)+\frac{1}{2}(x_3+x_4)\\
\frac{1}{2}(y_1+y_4)+\frac{1}{2}(y_2+y_3) &= \frac{1}{2}(y_1+y_2)+\frac{1}{2}(y_3+y_4)
\end{aligned}$$

When looking at the midpoints, though, they are given by the equivalents:

$$\begin{aligned}
x_1+x_3 &= x_2+x_4\\
y_1+y_3 &= y_2+y_4
\end{aligned}$$

Now the components (u,v) of the velocities are *only* defined at the quadrilateral midpoints. Due to isoparametrics (same linear interpolation at the edges), the same kind of equations are valid for these function values too:

$$\begin{aligned}
u_1+u_3 &= u_2+u_4\\
v_1+v_3 &= v_2+v_4
\end{aligned}$$

The computer doesn't "know" anything about this "geometry" in parameter space. And thus must be "told" explicitly about it ! It will be demonstrated now that, apart from these two, also the equations of motion contribute two discretized equations with each of the linear quadrilateral elements, which makes four (4) in total.

Discretization

It has not been explained yet how the partial differential equations describing ideal fluid flow are to be discretized at a linear quadrilateral. But it has been suggested already that a finite difference method should be employed for this purpose. So maybe we'd better forget about pursuing Finite Element Methods any further. Or, following D.B Spalding, we will ignore *the siren voices from the "finite-element" champ* [4] , at least for the moment being. Instead, we are tempted to take a serious look, finally, at the methods which, without doubt, are the most succesfull in the field of Fluid Flow: so-called Finite Volume Methods. I consider the book by Patankar [5] as the most basic reference here. Did nobody recognize the *staggered grid* in this configuration of linear quadrilaterals, where velocities are localized at the midside nodes ? Anyway, according to the Finite Volume methodology, each of the Partial Differential Equations at hand must be integrated over a finite volume:

$$I_+ = \iint \left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right] dx\, dy \qquad I_- = \iint \left[\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right] dx\, dy$$

In 2-D, of course, we are talking about finite area's instead of finite volumes. Most of the time, these volume / area integrals are readily transformed into the much simpler surface / line integrals, by straightforward application of the well known theorems by Gauss and Green:

$$I_+ = - \oint v.dx - u.dy \qquad I_- = \oint u.dx + v.dy$$

In our case, we will take the finite volume which is formed by the (parent) quadrilateral, which is the one with nodal points at the vertices:

```
4 _____3_____ 3
 |           |
 |           |
4             2
 |           |
 |_____ _____|
1      1      2
```

Consequently, the line integrals must be evaluated as follows:

$$\oint v.dx - u.dy = \left(\int_{(1)}^{(2)} v.dx - \int_{(1)}^{(2)} u.dy \right) + \left(\int_{(2)}^{(3)} v.dx - \int_{(2)}^{(3)} u.dy \right)$$

$$+ \left(\int_{(3)}^{(4)} v.dx - \int_{(3)}^{(4)} u.dy \right) + \left(\int_{(4)}^{(1)} v.dx - \int_{(4)}^{(1)} u.dy \right)$$

Function behaviour is linear at the edges of such a quadrilateral. Take one of the above integrals as an example how they must be evaluated:

$$\int_{(1)}^{(2)} v.dx = \int_0^1 \left[v_1 + \xi(v_2 - v_1) \right] (x_2 - x_1)\, d\xi =$$

$$\left[v_1.\xi + \frac{1}{2}\xi^2 (v_2 - v_1) \right]_0^1 (x_2 - x_1) = \frac{1}{2}(v_1 + v_2)(x_2 - x_1) = v_1(x_2 - x_1)$$

Where the latter v_1 is to be specified at the midside node (1). Look how lucky we are ! Since the function values of u and v are actually *unknown* at the vertices, it would have been virtually impossible, namely, to evaluate them at some other place. Now add all the contributions together and finally suppose the result is zero:

$$\begin{aligned}\oint v.dx - u.dy = \; & [v_1(x_2-x_1) - u_1(y_2-y_1)] + [v_2(x_3-x_2) - u_2(y_3-y_2)] \\ + \; & [v_3(x_4-x_3) - u_3(y_4-y_3)] + [v_4(x_1-x_4) - u_4(y_1-y_4)] = 0\end{aligned}$$

In very much the same way, we find:

$$\begin{aligned}\oint u.dx + v.dy = \; & [u_1(x_2-x_1) + v_1(y_2-y_1)] + [u_2(x_3-x_2) + v_2(y_3-y_2)] \\ + \; & [u_3(x_4-x_3) + v_3(y_4-y_3)] + [u_4(x_1-x_4) + v_4(y_1-y_4)] = 0\end{aligned}$$

Where the coordinates are discretized at the four vertices, while the velocity components are discretized at the four midside nodes. Herewith, our set of four (4) equations, to be associated with a linear quadrilateral ideal flow element, is completed. Resulting in two "paralellogram" equations and two equations of motion. It will be demonstrated now that this set of 4 local equations is both necessary and sufficient, provided that the number of Degrees Of Freedom (DOFs) is limited to the minimum required: just two (2) velocity components at each of the midside nodes.

Mathematical Induction

In order to establish whether the linear quadrilateral could be suitable for our purpose, we just have to make a count of the equations and the unknowns. Our only care seems to be about matching the number of independent variables with the number of independent equations, resulting in an excercise which is taylored to the producing power of an individual. Assume a grid of I times J of our linear quadrilaterals. And let's tabulate the results:

number of equations bulk	=	$2I.J$
the additional equations	=	$2I.J$
number of boundary conditions	=	$2I + 2J$
All summed together	=	$4I.J + 2I + 2J$
Total number of unknowns	=	$2\{(I+1)J + I(J+1)\}$

The number of equations is thus *exactly equal* to the number of unknowns. Now we know that our least squares finite element method is equivalent with such a system of finite difference equations. Therefore it may be concluded that the least squares minimum of the former is merely zero from the start. But, can it be assured that the abovementioned balancing is also valid for irregular, non-rectangular meshes ? That's a valid question, indeed.

```
 ___________________________
|      |      |      |      |
|      |      |      |      |
|______|______|______|______|______
       |      |      |      |      |
       |      |      |      |      |
       |______|______|______|______|
       |      |      |      |      |
       |      |      |      |      |
 ______|______|______|______|______|
|      |      |      |      |      |
|      |      |      |      |      |
|______|______|      |______|______|
```

In order to answer the question - in an affirmative sense - *mathematical induction* will be applied to the balancing. First we prove that the system is balanced for just *one* element. This is a rather trivial task. (It can eventually be inferred from the above for $I = 1$ and $J = 1$.) As a next step, assume that the system is balanced for M elements. Then prove that it is also balanced for $(M + 1)$ elements. The proof is then completed by induction to the number of elements.

Let's look at the details. Consider a mesh entirely consisting of quadrilateral elements with nodal points in the middle of the edges. Each of the elements corresponds with 4 discretized equations and 8 unknowns. The elements are joined together at the nodal points. Just one boundary condition is present at each of the edges at boundary.

Theorem. *For all possible configurations of the linear quadrilaterals, used for discretizing the Ideal Flow problem, the number of equations, generated by these elements, is equal to the number of unknowns.*

Proof. The theorem is certainly true for a mesh consisting of just one element, since then the number of unknowns (8) equals the number of equations (4) plus the number of boundary conditions (4).

Now suppose that the theorem is true for a mesh consisting of M elements. Then attach another quadrilateral element to that mesh. Four cases - topological equivalents - are distinguished for the place where the new element can be attached: four boundary edges, three boundary edges, two boundary edges, one boundary edge. A figure says more than a thousand words:

```
\_______/     \_______/      \              \
|       |     |              |              |
|  4.   |     |   3.         |  2.          |  1.
|       |     |              |              |
|_______|     |_______       |_______       |
/       \     /       \      /       \      /
```

The proof now proceeds as follows:
4. Four boundary edges. Means that 4 boundary conditions are replaced by 4 bulk equations
$= -4 + 4 = 0$: Balanced
3. Three boundary edges. Means that 3 boundary conditions are replaced by 4 bulk equations and 2 additional unknowns and 1 new boundary condition
$= -3 + 4 - 2 + 1 = 0$: Balanced
2. Two boundary edges. Means that 2 boundary conditions are replaced by 4 bulk equations and 4 additional unknowns and 2 new boundary conditions
$= -2 + 4 - 4 + 2 = 0$: Balanced
1. One boundary edge. Means that 1 boundary condition is replaced by 4 bulk equations and 6 additional unknowns and 3 new boundary conditions
$= -1 + 4 - 6 + 3 = 0$: Balanced
Thus if a mesh consisting of M elements is balanced then any mesh consisting of (M+1) elements also will be balanced. Since the mesh consisting of just one element is balanced, it follows - by mathematical induction - that all meshes consisting of any number of such quadrilateral elements always will be balanced.

Least Squares

In the previous paragraphs it has been established that the employment of our linear quadrilateral fluid flow element indeed leads to an algebraic system, consisting of N equations and N unknowns. Thus it seems that we are almost finished. Because the only thing that remains to be done is: to eliminate the unknowns in that algebraic system. Gaussian elimination, for example, could be employed for that purpose. Again, this sounds more trivial than it actually is. At first, the elementary equations found will be repeated here for convenience. Paralellogram:

$$u_1 - u_2 + u_3 - u_4 = 0 \quad \text{and} \quad v_1 - v_2 + v_3 - v_4 = 0$$

Incompressible:

$$\begin{aligned} &-(y_2 - y_1)u_1 + (x_2 - x_1)v_1 - (y_3 - y_2)u_2 + (x_3 - x_2)v_2 \\ &-(y_4 - y_3)u_3 + (x_4 - x_3)v_3 - (y_1 - y_4)u_4 + (x_1 - x_4)v_4 = 0 \end{aligned}$$

Irrotational:

$$\begin{aligned} &+(x_2 - x_1)u_1 + (y_2 - y_1)v_1 + (x_3 - x_2)u_2 + (y_3 - y_2)v_2 \\ &+(x_4 - x_3)u_3 + (y_4 - y_3)v_3 + (x_1 - x_4)u_4 + (y_1 - y_4)v_4 = 0 \end{aligned}$$

The main difficulty is in the fact that our system of equations seems *not* to be structured very well. This becomes readily apparent if we put our finite element difference equations in matrix form. In order to save space on paper, a few abbreviations are proposed:

$$x_{ji} = (x_j - x_i) \quad \text{and} \quad y_{ji} = (y_j - y_i)$$

Then the global matrix is formed by simply repeating $N/4$ times the kind of rows shown here:

$$\begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 \\ -y_{21} & x_{21} & -y_{32} & x_{32} & -y_{43} & x_{43} & -y_{14} & x_{14} \\ x_{21} & y_{21} & x_{32} & y_{32} & x_{43} & y_{43} & x_{14} & y_{14} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix}$$

It is always desirable that especially the matrix elements needed for *pivoting* can be found quickly and reliably, independent of the question whether a direct or an iterative solution method is to be preferred. With many Finite Difference and Finite Element methods, the pivoting elements are always at the main diagonal of the system matrix. Most of the time, the matrices obtained are even positive definite. With such system matrices, intuitively, an equation with a certain rank number is designated to an unknown with the same ordinal number. We could call such systems, where each equation so to speak "belongs" to an unknown: *to-the-point*. Good examples of these to-the-point systems are provided by the common Convection - Diffusion problems. Let it be emphasized from the start that *the Least Squares Finite Element Method doesn't contribute anything valuable for systems that are to-the-point already.*
Not so with the above bare system of F.D. equations for Ideal Fluid Flow. It is not evident at all how an equation should be attached to an unknown. None of these equations seems to "belong" to an unknown. They are thus certainly not to-the-point. And I have no idea where the pivots are.

This is precisely the place where the Least Squares Finite Element Method comes into the picture again. Because what will happen if we just square the finite difference equations and put the result to zero ? Since abstraction is the key to generalization, let's write the above as 4 equations with 8 unknowns, apart from any distracting details:

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} & A_{17} & A_{18} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} & A_{27} & A_{28} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} & A_{37} & A_{38} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} & A_{46} & A_{47} & A_{48} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{bmatrix}$$

When compared with matrix notation, the summation notation is much more compact, and closer to programming practice too:

$$\sum_{j=1}^{8} A_{i,j}\, w_j \quad \text{for} \quad i = 1, \ldots, 4$$

Now square the equations and let the result be vanishing:

$$\sum_{i=1}^{4} \left(\sum_{j=1}^{8} A_{i,j}\, w_j \right)^2 = \text{minimum}(w_j) = 0$$

A necessary condition for minimizing a function is that the partial derivatives to its parameters become zero:

$$\frac{\partial}{\partial w_i} \left[\sum_{k=1}^{4} \left(\sum_{i=1}^{8} A_{k,i}\, w_i \right) \left(\sum_{j=1}^{8} A_{k,j}\, w_j \right) \right] = 0 \quad \Longleftrightarrow$$

$$\sum_{j=1}^{8} \left[\sum_{k=1}^{4} A_{k,i}\, A_{k,j} \right] w_j = 0 \quad \text{for} \quad i = 1, \ldots, 8$$

Or:

$$\sum_{j=1}^{8} [E_{i,j}]\, w_j = 0 \quad \text{for} \quad i = 1, \ldots, 8 \qquad \text{where} \quad E_{i,j} = \sum_{k=1}^{4} A_{k,i}\, A_{k,j}$$

The name E is not chosen by coincidence. A closer look at the formulas reveals, namely, that the coefficients $E_{i,j}$ behave exactly *as if* they were the coefficients of a 8×8 Finite Element matrix. This element matrix is formed, though, by multiplying the 8×4 transpose of a Finite Difference matrix with the 4×8 original F.D. matrix: $E = A^T A$.

A small modification of the above should be notified. It is advantageous to carry out the least squares assembly procedure with just *one* finite difference equation at a time, instead of all in once. If we adhere on this, then the least squares finite element matrix can be considered as a *superposition* of four (4) finite element matrices, one for each of the F.D. equations:

$$E_{i,j}^{(k)} = A_{k,i} A_{k,j} \quad \Longrightarrow \quad E_{i,j} = \sum_k E_{i,j}^{(k)}$$

The advantage of this modification will become clear in the next paragraph, on *Trace Weighting*: it is desirable, namely, that each of the F.D. equations can be weighted separately.

An assembly procedure for finite element matrices is notoriously different from an assembly procedure for finite difference matrices. The latter are simply plugged in, one after another, line by line. But the coefficients of an element matrix must be added to previous contributions in the global system, which, as a rule, requires somewhat more skillful coding.
Apart from such technical details, however, what shall be considered as the main achievement of the least squares method ? Let's start with the remark that an L.S. system matrix will always be *symmetric* and *positive definite*. Moreover, it will become obvious that the k'th equation in the system indeed belongs to the k'th unknown and that all pivots are to be found at the main diagonal. So the least squares finite element difference method surely exhibits a number of desirable properties. It's very well to-the point. That is the good news. The bad news is that there are some possible drawbacks too. These will be taken care about in the next paragraph.

Trace Weighting

It is known from litterature that the Least Squares Finite Element "variational integral" can be modified a little bit as follows, according to Zienkiewicz [3] chapter 3.14.2 equation (3.168):

$$\iint \left\{ A.\left[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right]^2 + B.\left[\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right]^2 \right\} dx.dy = \text{minimum}$$

Here A and B are "positive valued functions or constants", which may be chosen in a convenient way. Quoting without permission from Zienkiewicz: *Once again this weighting function could be chosen as to ensure a constant ratio of terms contributed by various elements - although this has not yet been put into practice.* Well, somebody has to be the first ...
Suppose that we plan to adhere to this practice indeed. Then it should be remarked immediately that, for example, the area of an element is not relevant anymore: it is "absorbed" into the constants A and B, which are arbitrary. So constants like J and w may be left out altogether; appropriate weighting factors may be chosen differently instead. Let's jump to the discretized equivalent of the above:

$$\sum_{k=1}^{4} \left(\sum_{j=1}^{8} A_{k,j}\, w_j \right)^2 = \text{minimum}(w_j) = 0$$

Ziekiewicz' weighting procedure translates into the notion that coefficients of the finite difference matrix rows A_k are determined up to an arbitrary constant. This means that it is admissible to multiply every row with a certain (positive) number α_k, for the purpose of optimizing something:

$$\sum_{k=1}^{4} \left(\sum_{j=1}^{8} \alpha_k \, A_{k,j} \, w_j \right)^2 = \mbox{minimum}(w_j) = 0$$

This additional degree of freedom is a good thing, because herewith a remedy may be found for still another tricky phenomenon, associated with L.S.FEM: taking the square of a system of equations will also square the *condition number* of that system. I'm not going to explain why the condition number of a matrix is the quotient of its greatest and its smallest eigenvalue. Some decent references can be found on the Web:

`http://www.math.gatech.edu/~bourbaki/math2601/Web-notes/1num.pdf`

Now, if the eigenvalues of the original matrix are given by λ, then the eigenvalues of the squared matrix (that is: the transpose multiplied by the original) are given by the squares of the absolute values of the same λ:

$$A\,x = \lambda\,x \quad \Longrightarrow \quad A^T A\,x = \lambda^* \lambda\,x = |\lambda|^2\,x$$

The condition number $C(A)$ of a matrix A is defined as the absolute value of its largest eigenvalue $\lambda_{max}(A)$ divided by its smallest eigenvalue $\lambda_{min}(A)$. It can be demonstrated that the condition number is kind of a measure for the loss of decimals with operations like inverting the matrix. Therefore a large condition number is considered to be bad. It is easily shown that the condition number of a squared system of equations is quadratically *worse*, when compared with the condition number of the original system:

$$C(A) = \frac{\lambda_{max}}{\lambda_{min}} \quad \Longrightarrow \quad C(A^T A) = \left| \frac{\lambda_{max}}{\lambda_{min}} \right|^2$$

Hence it is clear that attention should be given to the condition of equations emerging from a least squares procedure. Putting the idea of Zienkiewicz into practice, a proper choice for the weighting factors α_k indeed should be employed for optimizing the condition of the global Least Squares matrix. It's common practice to arrange things in such a way that the contributions of all rows become approximately the same. Probably the easiest way to accomplish this would be: to divide every F.D. Equation (k) by its own "length". Meaning that $\alpha_k = 1/L_k$ in:

$$\sum_{j=1}^{8} \frac{A_{k,j}}{L_k} w_j \qquad \mbox{with} \quad L_k = \sqrt{\sum_i A_{k,i}^2} \qquad k = 1, \ldots, 4$$

Now take a look at the element matrix E which is associated with *one* of the equations (k) in the associated finite difference problem:

$$E_{i,j}^{(k)} = A_{k,i}\,A_{k,j} \quad \Longrightarrow \quad E_{i,i}^{(k)} = A_{k,i}^2 \quad \Longrightarrow \quad Sp\left[E^{(k)}\right] = \sum_i A_{k,i}^2 = L_k^2$$

Therefore the coefficients $\sum_i A_{k,i}^2$ are also found as the *trace Sp* of the element matrix belonging to one of the finite difference equations. Thus the weighting procedure can also be accomplished by dividing all coefficients of a "single" element matrix by the *trace* of this matrix. Hence the name: *trace weighting* or "SpoorWegen" (Pun: "RailRoads" in Dutch. And don't confuse weight tracing as opposed to ray tracing ... ;-)
How about the following little **theorem**: *The trace of the whole system's matrix is equal to the total number of finite difference equations involved.* Providing the programmer with a means to check out whether the requirement of balancing the (normed) F.D. equations is actually fulfilled.

Repeatable Elements

Our least squares finite elements for Ideal fluid flow are ideal in more than one sense. For example, as soon as they are rotated over a certain angle, or scaled up with a certain amount, then the accompanying finite element matrix may still be exactly the same. This effectively means that the element has to be calculated only *once* : it can be repeated. An immense advance at the slow PC's where these programs were developed, but still an elegant incidental with the nowadays abundance of memory and processing power. Let's investigate how repeatability happens to be there.

Least Squares Finite Elements are produced in our theory by taking the square of Finite Difference like equations:

$$E_{i,j}^{(k)} = A_{k,i}\, A_{k,j}$$

Here $A_{k,i}$, for $k = 3, 4$, are coefficients of the following equation system. (The "parallelogram" equations need not to be considered here.)

$$\begin{aligned}
&-(y_2 - y_1)u_1 + (x_2 - x_1)v_1 - (y_3 - y_2)u_2 + (x_3 - x_2)v_2 \\
&-(y_4 - y_3)u_3 + (x_4 - x_3)v_3 - (y_1 - y_4)u_4 + (x_1 - x_4)v_4 = 0 \\
&+(x_2 - x_1)u_1 + (y_2 - y_1)v_1 + (x_3 - x_2)u_2 + (y_3 - y_2)v_2 \\
&+(x_4 - x_3)u_3 + (y_4 - y_3)v_3 + (x_1 - x_4)u_4 + (y_1 - y_4)v_4 = 0
\end{aligned}$$

Suppose the problem is transformed by an orthogonal transformation, say a rotation over an angle α:

$$\begin{aligned}
x = cos(\alpha).x + sin(\alpha).y \quad &; \quad y = -sin(\alpha).x + cos(\alpha).y \\
u = cos(\alpha).u + sin(\alpha).v \quad &; \quad v = -sin(\alpha).u + cos(\alpha).v
\end{aligned}$$

Then the first term of the first equation becomes:

$$-(y_2 - y_1)u_1 + (x_2 - x_1)v_1 :=$$

$$-\left[-(x_2 - x_1).sin(\alpha) + (y_2 - y_1).cos(\alpha)\right]\left[+u_1.cos(\alpha) + v_1.sin(\alpha)\right] +$$

$$[(x_2 - x_1).cos(\alpha) + (y_2 - y_1).sin(\alpha)]\,[-u_1.sin(\alpha) + v_1.cos(\alpha)] =$$
$$-(y_2 - y_1).u_1.\left[cos^2 + sin^2\right] + (x_2 - x_1).v_1.\left[sin^2 + cos^2\right] +$$
$$+(x_2 - x_1).u_1.[+sin.cos - cos.sin] + (y_2 - y_1).v_1.[-cos.sin + sin.cos] =$$
$$-(y_2 - y_1).u_1 + (x_2 - x_1).v_1$$

And the first term of the second equation becomes:

$$+(x_2 - x_1)u_1 + (y_2 - y_1)v_1 :=$$

$$[+(x_2 - x_1).cos(\alpha) + (y_2 - y_1).sin(\alpha)]\,[+u_1.cos(\alpha) + v_1.sin(\alpha)] +$$
$$[-(x_2 - x_1).sin(\alpha) + (y_2 - y_1).cos(\alpha)]\,[-u_1.sin(\alpha) + v_1.cos(\alpha)] =$$
$$+(x_2 - x_1).u_1.\left[cos^2 + sin^2\right] + (y_2 - y_1).v_1.\left[sin^2 + cos^2\right] +$$
$$+(x_2 - x_1).v_1.[cos.sin - sin.cos] + (y_2 - y_1).u_1.[sin.cos - cos.sin] =$$
$$+(x_2 - x_1).u_1 + (y_2 - y_1).v_1$$

Conclusion: *both* the discretized equations of motion are *invariant* for rotations. Now it is trivial that they are also invariant for translations (: differences of all coordinates). So both these F.D. equations are independent of the observer's coordinate system as a whole.

Now we know from the "Trace Weighting" paragraph that L.S. elements allways can be normed. So multiplying the A_i with one and the same factor has no influence whatsoever upon the element-matrix. But coefficients A_i depend only upon coordinate-differences. Consequently, the element-matrix must remain the same if these distances are all multiplied with a constant factor. Therefore, two element-matrices must be the same if their element geometries only differ by a scaling factor. In case of Labrujère's problem, this can be accomplished as follows. Start with the circular cylinder at a radial position R_0 ; suppose the mesh ends at a radial position R_n . Let the mesh as a whole be given by:

$$x_{i,j} = R_i\,cos(\alpha_j) \quad ; \quad y_{i,j} = R_i\,sin(\alpha_j)$$

Where $\alpha_j = (j-1)/(J-1)\cdot\pi/2$ and $j = 1,\ldots,J-1$. Then we have the little **Theorem.** *A mesh consisting of repeatable elements is given by radial positions:*

$$R_i = R_0.\left(\frac{R_n}{R_0}\right)^{(i-1)/(I-1)} \qquad \textit{where:} \quad i = 1,\ldots,I$$

Proof. There are two edges of the quadrilaterals which have the same length, equal to:

$$R_{i+1} - R_i = R_0.\left[\left(\frac{R_n}{R_0}\right)^{i/(I-1)} - \left(\frac{R_n}{R_0}\right)^{(i-1)/(I-1)}\right] = R_i.\left[\left(\frac{R_n}{R_0}\right)^{1/(I-1)} - 1\right]$$

Let $\gamma = (\alpha_{j+1} - \alpha_j)$ and $\delta = \left[(R_n/R_0)^{1/(I-1)} - 1\right]$ for short. Then the lengths

And likewise for the streamfunction:

$$\left[\begin{array}{cc} 1 & -1 \\ -1 & 1 \end{array}\right] \left[\begin{array}{c} \psi_1 \\ \psi_2 \end{array}\right] = \left[\begin{array}{c} +u.(y_2 - y_1) - v.(x_2 - x_1) \\ -u.(y_2 - y_1) + v.(x_2 - x_1) \end{array}\right]$$

The matrices and vectors are assembled and solved in a finite element manner.

In order to be able to apply the reverse procedure, *both* the Potential and the Streamfunction must be calculated first. For the Ideal Flow problem at hand, this can be done by solving standard Laplace equations numerically, with help of a Finite Element procedure which is very much standard. The boundary conditions for these problems do not present any special difficulties either.
After solving the Laplace problem for *both* ϕ and ψ, finally the velocity field can be calculated, using the above in a reverse fashion, namely as two equations with two unknowns:

$$(x_2 - x_1).u + (y_2 - y_1).v = \phi_2 - \phi_1$$
$$(y_2 - y_1).u - (x_2 - x_1).v = \psi_2 - \psi_1$$

The solution (u, v) of this system is:

$$u = \frac{(\psi_2 - \psi_1)(y_2 - y_1) + (\phi_2 - \phi_1)(x_2 - x_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$v = \frac{(\phi_2 - \phi_1)(y_2 - y_1) - (\psi_2 - \psi_1)(x_2 - x_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

It is remarked that the denominator in these expressions is always positive and

$$E = \begin{bmatrix} (y_2 - y_1)^2 & -(x_2 - x_1)(y_2 - y_1) \\ -(x_2 - x_1)(y_2 - y_1) & (x_2 - x_1)^2 \end{bmatrix} / \left\{(x_2 - x_1)^2 + (y_2 - y_1)^2\right\}$$

$$\text{and} \quad r = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Where E is the element matrix and r is the load vector (right hand side).

References

[1] Th.E. Labrujère,
'DE "EINDIGE ELEMENTEN - KLEINSTE KWADRATEN" METHODE TOEGEPAST OP DE 2D INCOMPRESSIBELE STROMING OM EEN CIRKEL CYLINDER', Memorandum WD-76-030, Nationaal Lucht- en Ruimtevaartlaboratorium (NLR), Noordoostpolder, 23 februari 1976.

[2] G. de Vries, T.E. Labrujère, D.H. Norrie,
'A LEAST SQUARES FINITE ELEMENT SOLUTION FOR POTENTIAL FLOW', Report No.86, Department of Mechanical Engineering, The University of Calgary, Alberta, Canada, December 1976.

[3] O.C. Zienkiewicz,
'The Finite Element Method', 3th edition, Mc.Graw-Hill U.K. 1977, ISBN 0-07-084072-5

[4] D.B. Spalding,
'A GENERAL PURPOSE COMPUTER PROGRAM FOR MULTIDIMENSIONAL ONE- AND TWO-PHASE FLOW', Imperial College, London SW7 (Paper to be presented at IMACS meeting at Lehigh University, 1981)

[5] S.V. Patankar,
'Numerical Heat Transfer and Fluid Flow', Hemisphere Publishing Company U.S.A. 1980