Low-Velocity Flows in Constructed Wetlands: Physico-Mathematical Model and Computer Codes in Matlab-Environment

Låg-hastighetsflöden i konstruerade våtmarker: Fysikal-matematisk modell och programkoder i Matlab-miljö

Anders Wörman

Johan Kjellin

SLU
Institutionen för biometri och teknik

Rapport – miljö, teknik och lantbruk 2006:01

SLU
Department of Biometry and Engineering

Uppsala 2006
ISSN 1652-3237

Tryck: SLU Service/Repro, Uppsala 2006

ABSTRACT
Constructed wetlands play an essential role in reducing nutrient content and heavy metals in wastewater and run-off water. The treatment efficiency strongly depends on flow pattern and residence times of the water, and therefore further understanding of water flow in constructed is of great importance. Flow modeling is here an efficient tool, and this report describes a 2D model for water flow in constructed wetlands. The model is implemented in Matlab environment and applies for stationary, "low velocity" flow with free surface and small water depth relative to the horizontal extension of the wetland. Simulations compute the elevation of the free water surface and calculate flow velocities. Furthermore, a particle tracking routine gives information of flow paths and residence times of the wetland, where particles can be subjected to both dispersion and exchange with stagnant zones. The main purpose of the model is to provide fast and useful computations for practice in wetland technology, and especially to adapt the mathematical formulation to the specific flow conditions prevailing in wetlands.

SAMMANFATTNING
Konstruerade våtmarker blir allt vanligare för att reducera halten av näringsämnen och tungmetaller i avrinningsvatten från jordbruksmarker samt avloppsvatten. Hur pass väl reningen fungerar beror i stor utsträckning på vattnets flödesmönster och uppehållstider i våtmarken. Därför är det väsentligt att förstå hur man påverkar dessa faktorer vid konstruktion av våtmarker. Till detta är flödesmodellering ett effektivt verktyg, och den här rapporten beskriver en 2D flödesmodell utvecklad för modellering av vattenflöde i våtmarker. Modellen är implementerad i Matlab miljö och är utvecklad för simulering av stationärt flöde med låga hastigheter i våtmarker med öppen vattenyta, och där våtmarkens horisontella utsträckning är stor i förhållande till vattendjupet. I modellen beräknas vattenytans nivå, samt flödeshastigheter. Dessutom finns en partikelspårningsfunktion som ger information om flödesmönster och uppehållstidsfördelning. Här kan även dispersion och utbyte med stagnanta zoner, såsom sediment och täta vegetationspartier, simuleras. Modellen är i första hand utvecklad som ett effektivt hjälpmedel för att utvärdera hur konstruktionen av våtmarker påverkar vattenflödet.

PREFACE

Constructed wetlands are used as polishing steps in municipal wastewater treatment. Observations indicate that the nitrogen load is reduced with about 40% in these systems, despite the fact that certain other studies indicate that these have far better capacity. This is a strong motivation for studying these systems in more detail and, especially, with the aid of mathematical models for representing the through flow of water.

This report describes a model of the water flow in constructed wetlands originally developed within the PRIMROSE project that is financed by the European Commissions fifth framework programme (Wörman, 2002). Further, development was undertaken of the model within a project financed by the Swedish Research Council for Environment, Agricultural Sciences and Spatial Planning (FORMAS).

Uppsala, April 2006

Anders Wörman

Johan Kjellin

TABLE OF CONTENTS

1. INTRODUCTION

This report describes a model of the water flow in constructed wetlands originally developed within the PRIMROSE project that is financed by the European Commissions fifth framework programme (Wörman, 2002). Further, development was undertaken of the model within a project financed by the Swedish Research Council for Environment, Agricultural Sciences and Spatial Planning (FORMAS).

The model is implemented in a matlab environment and is intended to apply for stationary, "low velocity" flow with free surface and small water depth relative to the horizontal extension of the wetland. Typical for open flows in constructed wetlands is low flow velocities to the extent that general formulations of the water motion (e.g. Saint Venant equations) are more complicated than really required for this special case.

The water flow in a wetland can be due to groundwater motion (sub-surface wetland) or overland flow (pond type wetland). The crucial factor for the applicability of the current model is that the flow is stationary and its velocities are sufficiently low, which they normally are in constructed wetlands (see section 2.1). The model simulates the elevation of the free water surface and calculates flow velocities. A particle tracking routine gives information of flow paths and residence times of the wetland. Particles can be subjected to both dispersion and exchange with stagnant zones.

The main purpose of the model proposed here is to provide fast and useful computations for practice in wetland technology and especially to adapt the mathematical formulation to the specific flow conditions prevailing in wetlands. This report describes the physical background and mathematical simplifications in the model development as well as computer codes implemented in a matlab environment. The physical background and limitation of the model is described in section 2 and a simple user manual for the codes is provided in section 3.

2. DESCRIPTION OF MATHEMATICAL MODEL

2.1 Physical background

The depth averaged form of the two equations describing free surface water flow is often referred to as Saint Venant equations after their inventor. The depth averaged form of the momentum equation for water flow in two dimensions (in the horizontal plane) can be stated as (Chow, 1959)

$$\frac{1}{g}\frac{\partial \mathbf{V}}{\partial t}+\frac{1}{g}\mathbf{V}\cdot\nabla\mathbf{V}+\nabla(h+z)=-\frac{f}{8R_h}\frac{\mathbf{V}|\mathbf{V}|}{g} \qquad (1)$$

where V = flow velocity (m/s), bold face type denote a vector quantity, g = acceleration due to gravity (m/s^2), h = water depth (m), z = elevation of bottom of flow in relation to an

Table 1: Typical hydraulic characteristics of two large constructed wetlands in Sweden.

	Area [m^2]	Mean depth [m]	Volume [m^3]	Discharge [l/s]	Mean residence time [h]	Mean path length [m]	Mean velocity [m/s]	Re = U h/ν [-]	$U^2/2g$ [m]
Ekeby wetland; pond 1 (first step)	**35 585**	**~0.93**	**~33 094**	**97 (mean over the year)**	**94.8 (Vol/Q)**	**363**	**$1.06x10^{-3}$**	**985**	**$60x10^{-9}$**
Alhagen wetland; Starrträsk and Vassträsk	**148 970**	**< 1**		**43 – 112 (December 2000)**	**148.56 (tracer test)**	**624**	**$1.16x10^{-3}$**	**1160**	**$70x10^{-9}$**

arbitrary (but constant) datum plane (m), f = Darcy Weisbach friction factor (-), R_h = hydraulic radius of the flow channels (m) and the Nabla operator $\nabla = (\partial/\partial x, \partial/\partial y)$ (in two dimensions). For groundwater flow the hydraulic radius corresponds to the size of the pore channels.

The second field equation needed to obtain a closed form formulation is the continuity equation

$$S\frac{\partial h}{\partial t} - \nabla \cdot (\mathbf{V}h) = 0 \qquad (2)$$

in which S = storativity (or volume of pores relative to total volume). For an open flow, S = 1.

For **stationary flow** we can neglect the time derivatives of Eqns. (1) and (2). Further, if $\mathbf{V} \cdot \nabla \mathbf{V} << \nabla(h + z)$, the equations can be even more simplified. This condition can be approximated by $\text{grad}(V^2/(2\,g)) << \text{grad}(h + z)$ or simply $V^2/(2\,g) << h$. Hence, by comparing the change in the velocity head or the magnitude of the velocity head relative to the gradient in water depth or the water depth, it is possible to judge if the **flow velocity is sufficiently low**.

Table 1 shows typical hydraulic quantities of two large constructed wetlands in Sweden, those at Ekeby and Alhagen. The velocity head is in the order of tens of nm (nano meters), which is negligible small with respect to the energy balance of the flow. For similar cases, for which the velocity head can be neglected, Eqn. (1) can be written for stationary flow

$$\nabla(h + z) = -\frac{f}{8R_h}\frac{\mathbf{V}|\mathbf{V}|}{g} \qquad (3)$$

This equation corresponds to Manning equation in which there is a non-linear relationship between bed slope and velocity for a uniform flow (V and h do not change with distance).

Generally, the friction factor can be expressed as a function of Reynolds number , Re = (Vh)/ν, and relative roughness, ε/h; f(Re ; ε/h), where V = | **V** |, ε = length scale of micro-topography of the bed and spacing of vegetation or other obstructions [m]. i.e., the roughness, and ν = kinematic viscosity [m^2/s]. Power functions have been applied to wetland flows (Bolster and Saiers, 2002) and here the friction factor is expressed in the form of

$$f = \alpha \left(\frac{\varepsilon}{h}\right)^{-m} \mathrm{Re}^{-n} \tag{4}$$

where α = a constant [-], n = a coefficient [-] and m = a coefficient [-].

A substitution of $f = \alpha\,(\varepsilon/h)^{-m}\,\mathrm{Re}^{-n}$ in (2) yields

$$h^{1+m+n}\nabla(h+z) = -FV^{1-n}\mathbf{V} \tag{5}$$

$$F = \frac{\alpha \varepsilon^m \upsilon^n}{2g} \tag{6}$$

For a laminar overland and sub-surface flow it has been found that n = 1, whereas n approaches zero as the flow becomes turbulent. As can be seen in Table 1, the average Reynolds number is in the order of 1000, which indicates that the flow is laminar on the average. Further, in a porous medium flow, like in a dense vegetation, m = -2. In terms of hydraulic conductivity for a porous medium flow, K [m/s], we have

$$K = 1/F \tag{7}$$

Bolster and Saiers (2002) found for a part of the Everglades wetland with laminar flow (i.e., they found that n = 1) that the coefficient m = -1.46.

If (5) is simplified for laminar flow (n = 1) and combined with the continuity equation for two dimensional stationary flow,

$$\nabla(\mathbf{V}h) = 0 \tag{8}$$

we obtain

$$\nabla \frac{h^{3+m}}{F}\nabla(h+z) = \nabla\left(\frac{h^{(3+m)}}{F}\nabla h\right) + \nabla\left(\frac{h^{3+m}}{F}\nabla z\right)0 \tag{9}$$

In wetland Alhagen, the bed surface elevation is fairly constant. Hence, we assumed that $\nabla(z) = 0$ as well as dense vegetation and, therefore, (9) is a linear function of h^{4+m}. Thus, if the second term in Eqn. (9) can be neglected we have a linear problem in $h^{(4+m)}$ and can use a direct solution technique. When bed surface elevation is not constant, $\nabla z \neq 0$, an iteration technique is needed to solve Eqn. (9), described in section 2.2.

A possibility is to recognise that m = -2 for porous medium flow and in (9) replace (3+m) with (1+nx), i.e. nx = m+2, where nx is another coefficient that adopts the value zero for porous medium flow.

A summary of the criteria required to be fulfilled in order for Eqn. (5) to apply is:

- Flow is stationary
- $\mathrm{Re} = V R_h/ < \sim 2000$
- $\mathbf{V}\cdot\nabla\mathbf{V} << \nabla(h+z)$, or as an approximation $V^2/(2g) << h$

2.2 Numerical methods

General

This version of matsolve.m solves Eqn. (9) for $\nabla z \neq 0$. An approximation of Eqn. (9) using central finite differences and can be expressed as

$$\frac{(\alpha_{j,i+1}+\alpha_{j,i})}{2\Delta X^2}h_{j,i+1} - \left[\frac{(\alpha_{j,i+1}+2\alpha_{j,i}+\alpha_{j,i-1})}{2\Delta X^2}+\frac{(\alpha_{j+1,i}+2\alpha_{j,i}+\alpha_{j-1,i})}{2\Delta Y^2}\right]h_{j,i} +$$
$$+\frac{(\alpha_{j,i}+\alpha_{j,i-i})}{2\Delta X^2}h_{j,i-1}+\frac{(\alpha_{j+1,i}+\alpha_{j,i})}{2\Delta Y^2}h_{j+1,i}+\frac{(\alpha_{j,i}+\alpha_{j-1,i})}{2\Delta Y^2}h_{j-1,i}+$$
$$\frac{(\alpha_{j,i+1}+\alpha_{j,i})}{2\Delta X^2}z_{j,i+1} - \left[\frac{(\alpha_{j,i+1}+2\alpha_{j,i}+\alpha_{j,i-1})}{2\Delta X^2}+\frac{(\alpha_{j+1,i}+2\alpha_{j,i}+\alpha_{j-1,i})}{2\Delta Y^2}\right]z_{j,i} +$$
$$+\frac{(\alpha_{j,i}+\alpha_{j,i-i})}{2\Delta X^2}z_{j,i-1}+\frac{(\alpha_{j+1,i}+\alpha_{j,i})}{2\Delta Y^2}z_{j+1,i}+\frac{(\alpha_{j,i}+\alpha_{j-1,i})}{2\Delta Y^2}z_{j-1,i}=0 \qquad (10)$$

where $\alpha = h^{1+nx}/F$, i denotes node number in x-direction (column in e.g. code and Numb) and j denotes y-direction (row in e.g. code and Numb). Eqn. (10) is also based on constant distances between the nodes.

The linear form of (10) implies that we can formulate the problem as

$$\mathbf{K_{mat}}\,\mathbf{h} + \mathbf{C_{mat}}\,\mathbf{z} = 0 \qquad (11)$$

Where **h** is the solution vector for **h**, $\mathbf{K_{mat}}$ is a coefficient matrix related to the h-terms of (10), **z** is a vector for the elevation of the bed and $\mathbf{C_{mat}}$ is a coefficient vector associated with **z**. The general principle is to identify one equation for each node point, which yields an exactly defined equation system. The only remaining problem is to identify "special" equations on the boundary.

Boundary condition

On the boundary, the five-point discretisation ([i,j], [i-1,j], [i,j-1], [i+1,j] and [i,j+1]) misses nodes outside the domain. This causes a need for special equations at the boundaries.

To recognise a constant hydraulic potential at a boundary node (von Neuman condition), we replace Eqn. (10) with

$$h+z = \text{known value} = \phi_0 \qquad (12)$$

This statement can be replaced with $h = \phi_0 - z$ = a known depth, which is the form utilised in the Matlab code.

Implementation of a known flow across the boundary (a known gradient in h+z, i.e. a Dirichlet condition) need to consider the stationary form of Eqn. (2): $Div(Vh) = Div(\alpha\nabla h + \nabla\alpha\nabla z)=0$. In a discrete form this becomes $(h\,V_x)_i$*DY-$(h\,V_x)_{i-1/2}$*DY+$(h\,V_y)_{j+1/2}$*DX/2-$(h\,V_y)_{j-1/2}$*DX/2 = 0, at the boundary on the right hand side, Fig 1. The "half-node-locations are

introduced because when V is expressed according to Eqn. (8) and h is formulated as central finite differences, only the node values are used to represent h. For instance, if the flux in terms of q = V h is known at i, the corresponding expression to Eqn. (10) with constant discretisation segments ΔX and ΔY becomes

$$\frac{q_{j,i+1/2}}{\Delta X} - \frac{\alpha_{j,i}+\alpha_{j,i-1}}{2\Delta X^2} h_{j,i}^{2+nx} + \frac{\alpha_{j,i}+\alpha_{j,i-i}}{2\Delta X^2} h_{j,i-1}^{2+nx} + \frac{\alpha_{j+1,i}+\alpha_{j,i}}{4\Delta Y^2} h_{j+1,i}^{2+nx} -$$
$$-\left[\frac{\alpha_{j+1,i}+\alpha_{j,i}}{4\Delta Y^2} + \frac{\alpha_{j,i}+\alpha_{j-1,i}}{4\Delta Y^2}\right] h_{j,i}^{2+nx} + \frac{\alpha_{j,i}+\alpha_{j-1,i}}{4\Delta Y^2} h_{j-1,i}^{2+nx} +$$
$$-\frac{\alpha_{j,i}+\alpha_{j,i-1}}{2\Delta X^2} z_{j,i}^{2+nx} + \frac{\alpha_{j,i}+\alpha_{j,i-i}}{2\Delta X^2} z_{j,i-1}^{2+nx} + \frac{\alpha_{j+1,i}+\alpha_{j,i}}{4\Delta Y^2} z_{j+1,i}^{2+nx} -$$
$$-\left[\frac{\alpha_{j+1,i}+\alpha_{j,i}}{4\Delta Y^2} + \frac{\alpha_{j,i}+\alpha_{j-1,i}}{4\Delta Y^2}\right] z_{j,i}^{2+nx} + \frac{\alpha_{j,i}+\alpha_{j-1,i}}{4\Delta Y^2} z_{j-1,i}^{2+nx} = 0 \qquad (13)$$

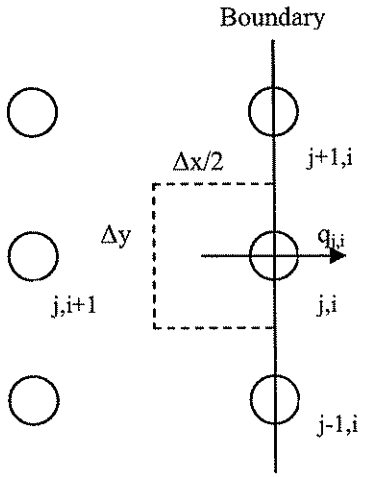

Fig 1. Boundary condition with known flow at the boundary on the right hand side.

The implication of this approach is that the flux across the boundary in any of the directions i+1/2, …j-1/2 can be explicitly expressed. For a closed boundary, the flux is zero.

2.3 Verification of numerical accuracy using closed-form solutions

A simple verification of the numerical accuracy of the solution to (9) was performed based on a one-dimensional solution. This comparison was based on the special case; nx=0, $\nabla z = 0$, a rectangular domain and a Dirichlet condition at two of the opposite (shorter) boundaries and no-flow conditions on the transversal boundaries. Basically the problem is one-dimensional with the closed-form solution

$$h^2(x) = (h_2^2 - h_1^2)\frac{x - x_0}{L - x_0} + h_1^2 \qquad (14)$$

where x is the coordinate in the flow direction, h_1 is the boundary condition at $x=x_0$ and h_2 is the boundary condition at x = L. As can be seen in Fig. 2 the deviation between the numerical and the closed-form solutions is marginal.

The other closed-form solution used to evaluate the calculation programme is that of a sink and a source placed in an infinite two-dimensional space, nx=-1 and $\nabla z = 0$ (cf. Fig. 3), which means that (9) is in the form of Laplace equation. The solution to this problem is readily obtained in cylindrical coordinates for one source at the time. The final solution in terms of velocity components is obtained by superimposing the two source solutions in Cartesian coordinates on the form of

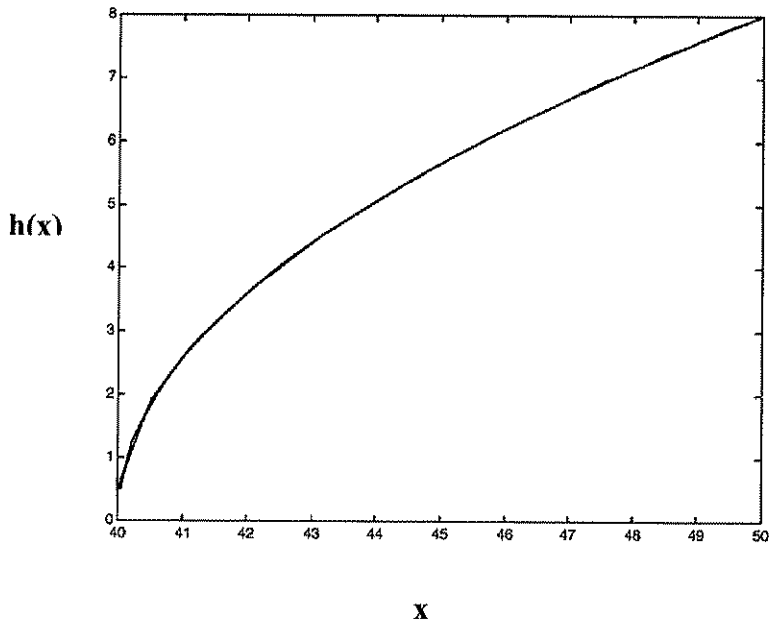

Fig. 2 Comparison of one-dimensional solution according to numerical model (solid curve) and closed-form solution (dashed curve).

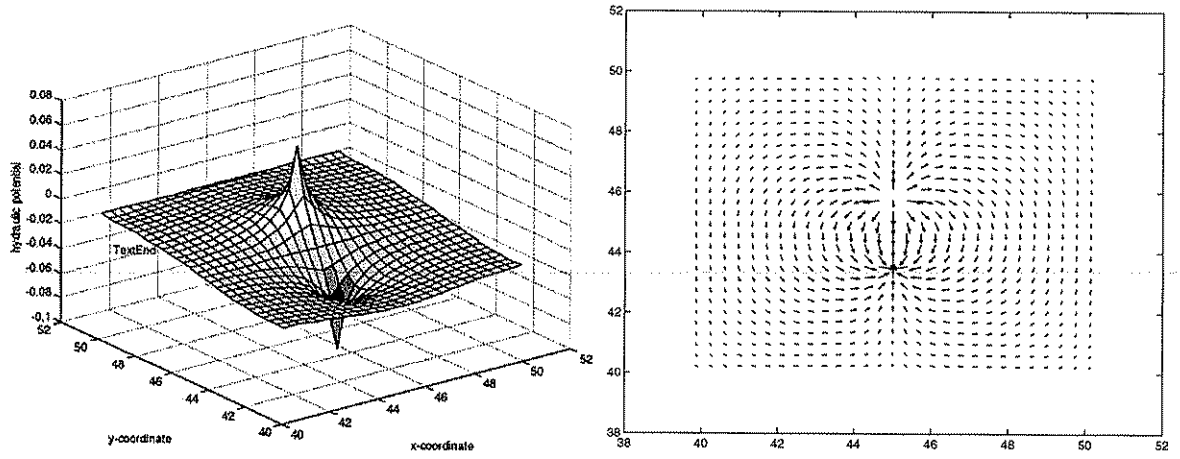

Fig. 3 Numerical solution to the problem with a source and a sink of equal strength. Potential surface is shown on the left-hand side and velocity vectors to the right.

$$v_x = S\left\{\cos\left[\arctan\left(\frac{x}{y}\right)\right]\frac{1}{x^2+y^2} - \cos\left[\arctan\left(\frac{x-L}{y}\right)\right]\frac{1}{(x-L)^2+y^2}\right\} \qquad (15a)$$

$$v_y = S\left\{\sin\left[\arctan\left(\frac{x}{y}\right)\right]\frac{1}{x^2+y^2} - \sin\left[\arctan\left(\frac{x-L}{y}\right)\right]\frac{1}{(x-L)^2+y^2}\right\} \qquad (15b)$$

where the origin is placed in the source point (in this example), S = strength of sources and L is the distance between the source and the sink.

The numerical solution is obtained for non-flow conditions at the boundaries and the problem formulation is consequently only an approximation of that stated for the closed-form solution. Therefore, the error was evaluated only at node points on node points in a square between the sink and the source. The mean error of the relative velocity deviation was 0.6% in one test.

3. SIMPLE USER GUIDE

This section contains a simple user guide and a description of programme features. Functioning of programme and definition of programme variables are described in section 4. Variables are marked with bold face type the first time they appear in their description.

In summary, the programme calculates the distribution of hydraulic potential or water surface elevation in two dimensions (horizontal plane) given a particular location of the boundary and statements of boundary conditions. Most statements are provided in indata files that are loaded into the matlab working space, but some statements need to be done directly in the executable files. All required files are listed in Appendix 1.

3.1 Indata

Before the programme is run, the problem needs to be defined in the following indata files (unformated text/ASCII-files):

koord: In this file the geometry of the calculation domain is defined. The real boundary is approximated by a limited number of straight lines that encloses the real domain (see section 4.1 for details).

BC1: This file states the type of boundary condition on each of the lines used to define the calculation domain. The programme can handle both a given flow (e.g. zero) or a known potential. Details are given in section 4.2.

BC2x and BC2y: These files state numerical values of the flux components or a given value of the potential. Details are given in section 4.2.

BC3: In this file it is possible to set boundary conditions to individual nodes used in the calculation. To set these conditions, it is necessary to first run the programme "flownet.m". When this programme has been run, the grid enumeration can be displayed and the specific nodes be identified. See section 4.2 for details.

F-values: Flow resistance or hydraulic conductivity, can be given explicitly in individual node points. Values are set in a matrix called "kdata". "kdata" should be loaded into the file "Conductivity.m", which is a helpfunction to "matsolve.m". Coordinates of the matrix "kdata" are given in "kkoord" in order to fit the data to the help grid system. If the flow resistance/hydraulic conductivity is constant (homogeneous condition), this can be set directly in the code of "matsolve.m". Details are given in sections 2.2 and 4.4.

nx-value: For open flow, the flow regime may be set directly in matsolve.m (see section 2.2 for details). For porous medium flow nx = 0. For parts of the Everglades nx=0.54.

zdata: In this file the bottom elevation is defined for the area. Details are given in section 4.3.

zkoord: This files states where the bottom elevation defined in zdata is placed in the help grid, fitting it to the same coordinate system as in koord. Details are given in section 4.3.

relinf: Release point of particles in the particle tracking routine is stated here, details are given in section 4.6.

Numerical variables may need to be set in the programme files (the matlab .m-files) before they are executed. The functioning of the programme and the different variables are described in section 4. The most essential variables that need to be set are:

NX: Is set in "flownet.m". Defines the number of segments used in the help grid (see section 4.2).

NY: Is set in "flownet.m". Defines the number of segments used in the help grid (see section 4.2).

transx: Is set in "flownet.m". Governs the placement of the help grid in relation to the approximated domain (see section 4.2).

transy: Is set in "flownet.m". Governs the placement of the help grid in relation to the approximated domain (see section 4.2).

fact: Is set in "flownet.m". Governs the placement of the help grid in relation to the approximated domain (see section 4.2).

bheta: Is set in "dispersion.m", which is a help function to "partrack.m". bheta defines the magnitude of dispersion in the particle tracking routine. bheta=0 means no dispersion and relevant values, evaluated in Ekeby treatment wetland with $\lambda_1=\lambda_2=0$, are 0.1 - 1.0.

λ_1, λ_2: Are set in "partrack.m". λ_1 is the probability for a water parcel to be immobilized in a stagnant zone. λ_1 is the probability for an immobile water parcel to be mobilized. $\lambda_1=0$ means no exchange with stagnant zones. Relevant values, evaluated in Ekeby treatment wetland with bheta=0, are $\lambda_1=\lambda_2=2.5*10^{-6}$ - $7.5*10^{-6}$.

3.2 Executing the programmes

Start by executing "flownet.m", which will lead to the generation of the calculation grid. After this has been run, it may be necessary to modify the grid generation by changing parameters directly in the matlab codes and execute flownet.m repeatedly. Analysis of the generated grid is needed in order to write the indata file BC3. In BC3, it is possible to assign boundary conditions to individual node points. See section 4.1 for details.

Run "BCpoints.m", but only if needed. This file loads BC3 and defines boundary conditions for individual node points. See section 4.2 for details.
Run "matsolve.m". This programme assembles the necessary matrices and solves the equation system described in sections 2.2 and 4.4 The hydraulic potential is contained in the matrix "pot". See section 4.5 for details.

The result can be analysed by running "velocities.m" and "partrack.m". "velocities.m" calculates velocity vectors and provides a plot of these vectors. "partrack.m" performs a

particle tracking and calculates certain characteristics, such as residence times of particles. The residence time distribution can also be analysed in more detail by running the code "PDFs.m".

4. DESCRIPTION OF CODES

4.1 Definition of boundary geometry and grid net: Load file "koord" and matlab-file "flownet.m"

The boundary position is defined in the text file "**koord**" (unformated). The first column of "koord" contains x-coordinates of points along the boundary and the second column contains corresponding y-coordinates:

```
64.81     35.19
70.37     46.30
68.52     64.81
...
```

In order to be able to construct a suitable grid net, the boundary can be translated in x- and y-directions using the variables "**transx**" and "**transy**" directly in the file "flownet.m" (Fig. 4).

On top of the approximated boundary, a help grid is defined with the vectors **XX** and **YY**. The numbers of segments in x- and y-direction are defined by **NX** and **NY**directly in

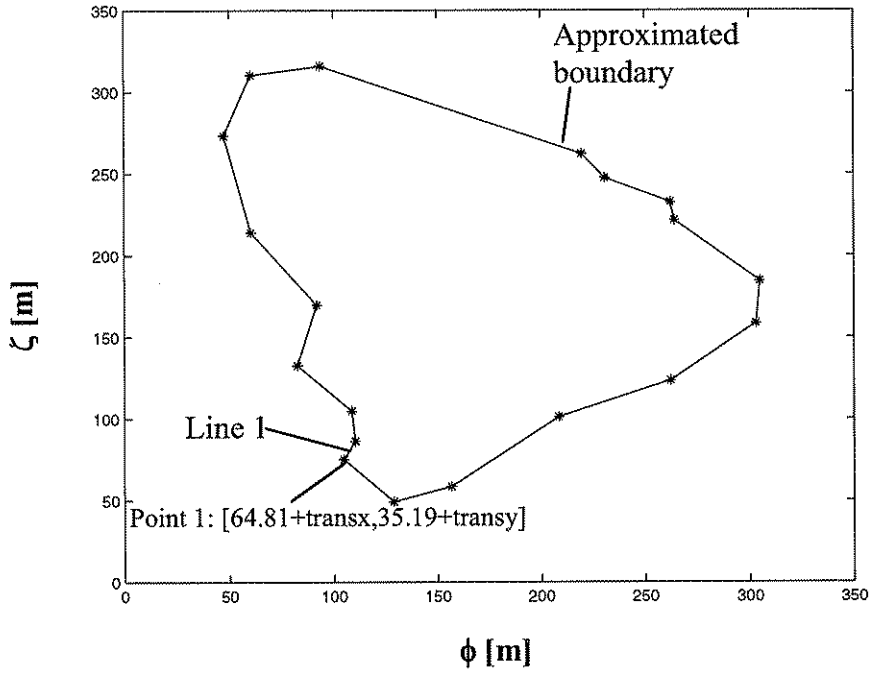

Fig. 4 Approximation of real boundary using discrete points on the boundary.

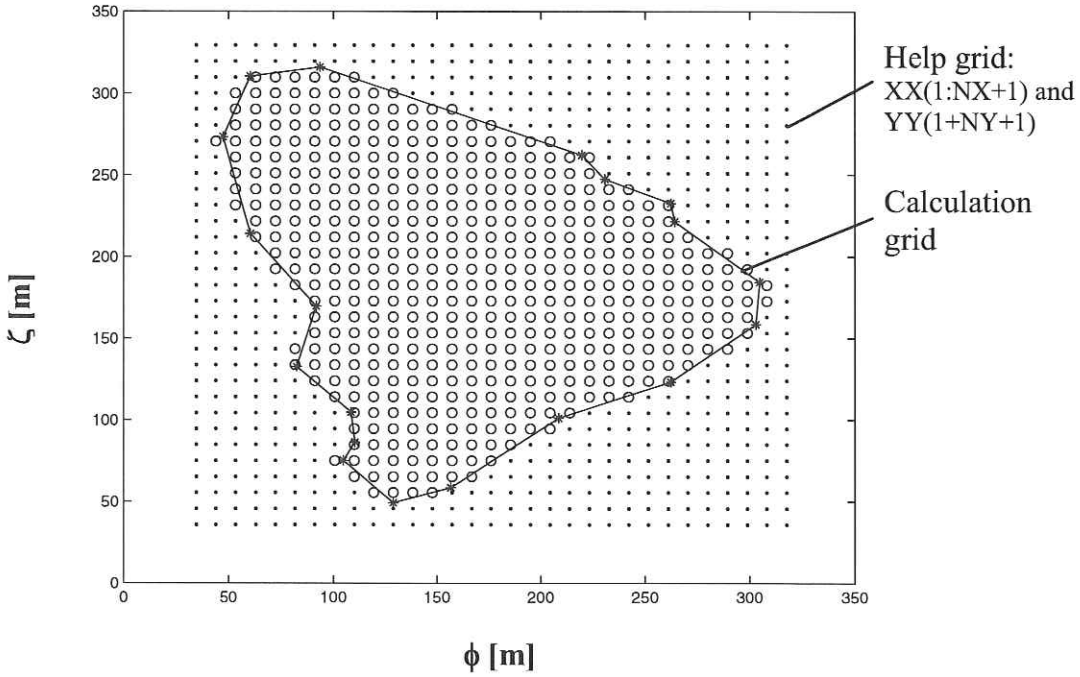

Fig. 5 Help grid net (small points and circles) in a 31 x 31 matrix and calculation grid (only circles). Number of segments in help grid is defined by NX and NY.

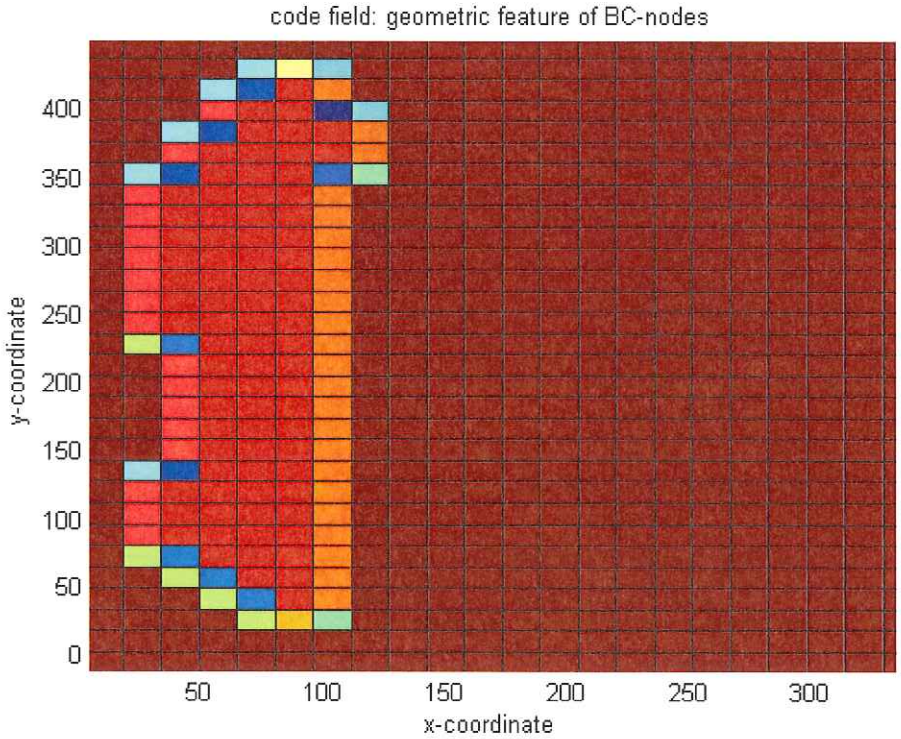

Fig. 6 Colour coding of grid points (each square corresponds to one grid node) with respect to their geometrical appearance. This information is stored in "code". Colour code is explain in text.

"flownet.m" (number of points are NX+1 and NY+1). One reason for introducing the help grid is that the matlab plot commands require rectangular fields (matrix arguments). The variable "fact" is used directly in "flownet.m" to enlarge the help grid (in percentage of the size of the approximation domain) as can be seen in Fig. 5. "flownet.m" uses equidistant node separation in x- and y-directions.

The file "flownet.m" analyses which help grid points are inside the approximated domain and assigns a numerical value to the matrix **code**(NX+1,NY+1) for each point included and zero for each point excluded from the calculation grid (Fig. 5). If there is a point in the help grid closer to the approximated boundary than that inside the domain on the same *row*, this node is included in the calculation grid.

Since the assemblage of the simultaneous equation system depends on the geometrical feature of the calculation grid at the boundary, the matrix "code" makes use of the following coding of the node points in the help grid (Figs. 6 and 7):

0 = Dark red in Fig. 6 = not included node
1 = Middle red in Fig. 6 = node included in grid
2 = Light red in Fig. 6 = boundary condition, node on vertical node-line with domain on the right hand side (three nodes in a column)
3 = Orange in Fig. 6 = boundary condition, node on vertical node-line with domain on the left hand side (three nodes in a column)
4 = Dark yellow (beige) in Fig. 6 = boundary condition, node on horisontal node-line with domain on higher-y-side (lower row!) (three nodes on a horisontal line)
5 = Light yellow in Fig. 6 = boundary condition, node on horisontal node-line with domain on lower-y-side (three nodes on a horisontal line)
6 = Green in Fig. 6 = boundary condition, corner node - lower-y-side (upper row!) and left
7 = Green-turquoise in Fig. 6 = boundary condition, corner node - lower-y-side and right
8 = Blue-turquoise No. 2 in Fig. 6 = boundary condition, corner node - higher-y-side and left
9 = Lightest blue in Fig. 6 = boundary condition, corner node - higher-y-side and right
10 = Light blue in Fig.6 = boundary condition, inner corner node – lower-y-side and left
11 = Middle blue in Fig. 6 = boundary condition, inner corner node – lower-y-side and right
12 = Dark blue in Fig. 6 = boundary condition, inner corner node – higher-y-side and left
13 = Darkest blue in Fig.6 = boundary condition, inner corner node – higher-y-side and right

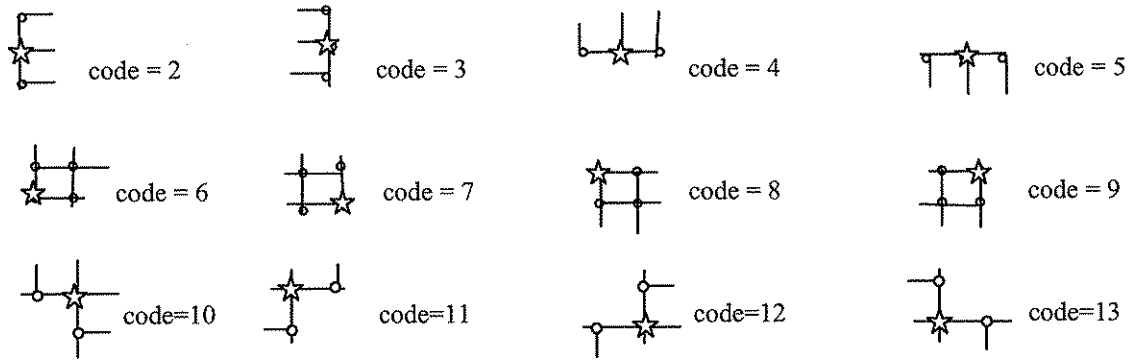

Fig. 7 Coding of grid nodes on the boundary(marked with star) with respect to structure of neighbouring nodes. This information is stored in "code".

Node with only one neighbouring node (as in Fig. 5 on lower left side) are omitted from the final help grid.

Note that increasing y-coordinate corresponds to higher rows in code(row,column). Hence, if code is displayed in the command window of matlab, a number "9" will appear on corner nodes on the lower right side. Fig. 6 shows the result of the following command series in "flownet":

```
surf(XX,YY,-code)
xlabel('x-coordinate'),ylabel('y-coordinate')
AZ = -17.5; EL = 90; view(AZ,EL)
```

Further, the calculation nodes are numbered in order from first row and first column, first row and second column, ... last row and last column in the matrix "**Numb**". This information is used when the coefficient matrix is assembled.

4.2 Definition of boundary condition: Load files "BC1", "BC2x", "BC2y" and "BC3" and matlab-files "flownet.m" and "BCpoints.m"

The boundary points are coded with respect to their boundary condition. The load file "BC1" is used to code the lines in the approximation domain. The following coding is used
1: von Neuman = specified value on water surface elevation
2: Dirichlet = specified flow in m^2/s) conditions
The actual definition of a Dirichlet condition is a known value of the derivative of the unknown function, but since the derivative of the hydraulic potential is linearly proportional to flow (in the Darcy case, see section 2), the condition is stated in terms of flow. BC1 should be arranged in a single column. In BC2x and BC2y, numerical values are assigned for the magnitude of the condition and sign for direction in the coordinate system XX and YY. Each file should be arranged as a single column.

A common situation for flow in a wetland is to have a known potential at the in- and outflow section and zero flow across the boundary in between. Hence, if BC1 states a "2" for line 1 (at first row in the single column of BC1) we know that line 1 associates with a flow condition. Further, by assigning "0" in the first row of both BC2x and BC2y we know that the x- and y-components of the flow is zero. A known leakage through the boundary in terms of V h (see section 2.2) should be stated in units [$m^3/(s\ m)$] in BC2x and BC2y. The sign of the flux should be defined in terms of the coordinate system, not e.g. direction in relation to the boundary. For instance, an inflow is positive on the "left-hand side" of the domain, whereas an inflow is negative on the "right-hand side".

A possibility is that the inflow section is given by a line defined in BC1 (the position of the line is given in "koord"). Hence, BC1 should state a "2" at the row corresponding to the line and the corresponding flux components in BC2x and BC2y.

The in- and outflow can be defined through the lines defined in koord by stating a known hydraulic potential in these sections. Place a "1" in BC1 at the row corresponding to the certain line and the value of the hydraulic potential in BC2x. In this case of a von Neuman condition, the content of in the file BC2y on the row corresponding to the certain line is ignored. Any numerical value can be (and also must be) stated on this row in BC2y.

The type of boundary condition is stored in "**code2**(NX+1,NY+1)" using the same indices as for "code" and "Numb". The numerical values of the grid points are stored in **code3x** and **code3y**.

A suitable way of defining the in- and outflow section to many wetlands is to define the hydraulic potential in individual node points. After "flownet.m" has been run, you can analyse the location of the calculation points in the XX(1:NX+1)-YY(1:NY+1)-system. One way to see the location where the in- and outflow should be defined is to use the plot of the matrix "code" that is provided by the "flownet.m" programme (cf. Fig. 6). It is also possible to display "code" in the command window row-by-row by writing "code(*row*,:)" followed by return.

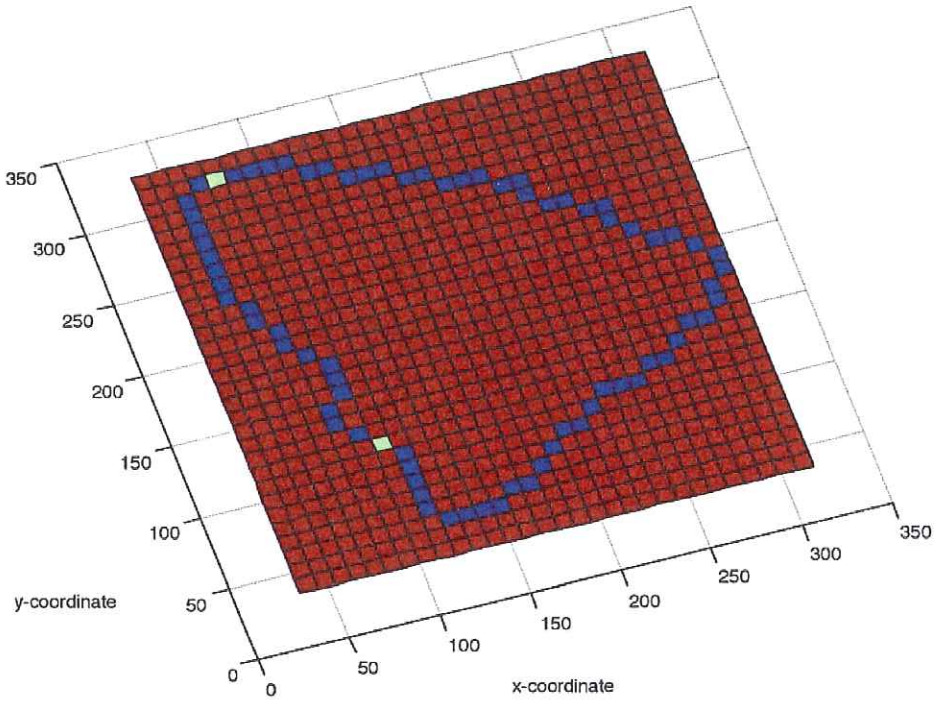

Fig. 8 Coding of boundary points with respect to their boundary condition, von Neuman and Dirichlet, respectively. This information is stored in "code2". Dark blue implies a zero flux and light green implies a known potential. The matrices code3x and code3y contain numerical values of the boundary conditions.

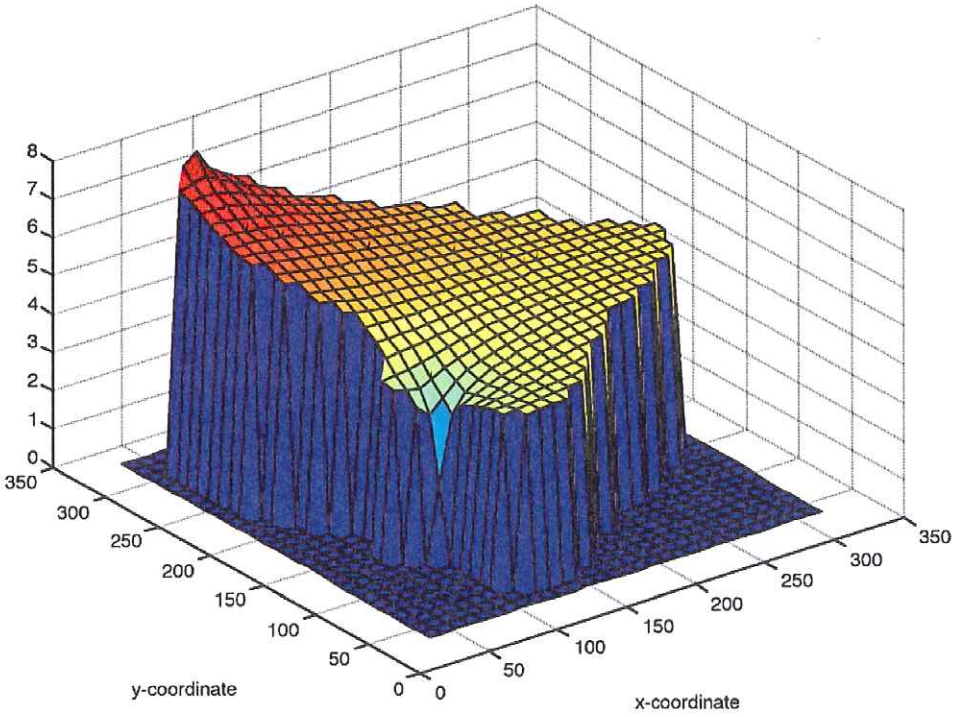

Fig. 9 Hydraulic potential or water surface as plotted by the routine "matsolve.m".

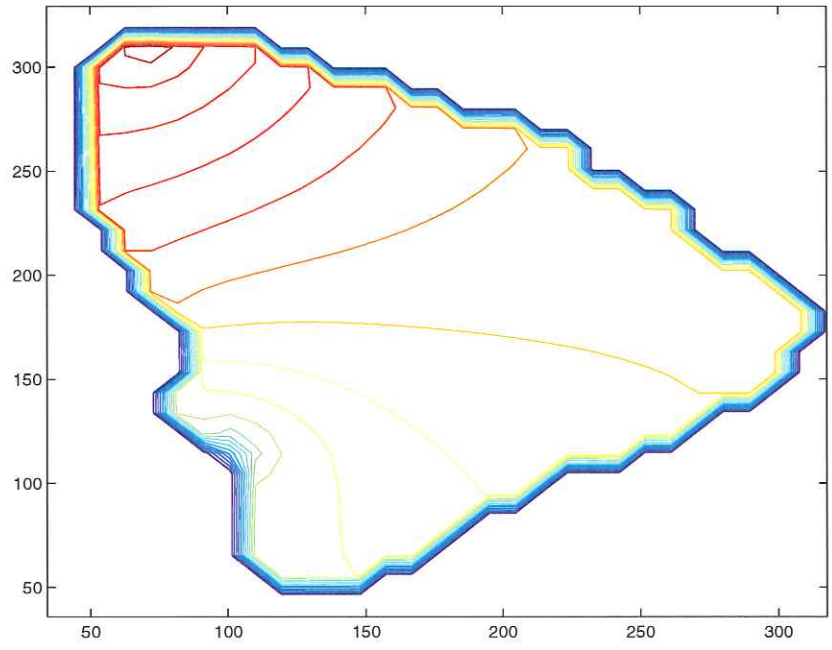

Fig. 10 Equipotential lines as plotted by the routine "matsolve.m".

When the right position has been identified the following information should be written in the file BC3:

Six columns define:
1,2) Node row resp. column number (in help grid)
3) BC type
4,5) If Dirichlet: Numerical value in x- respectively y-direction
If Von Neuman: Numerical value only in column 4.
6) If Dirichlet: Information whether there is inflow or outflow from the area in this node;
1 = inflow, 0 = outflow

This information is written on each single row for each node used in the definition. For the coding exemplified in Fig. 8, the BC3 file looks like this:

```
9           8           1           2           -           -
29          5           1           8           -           -
```

For a comparison of the column and row numbers, please confer Fig. 5 in which the rows and columns are easy to count.

4.3 Defining bottom elevation: Files "zdata" and "zkoord"

Bottom elevation is defined in the input file "zdata". In this file, the wetland area should be classified in a grid system, in which each node has a z-value representing the bottom elevation. The grid system has to be orthogonal. It also needs to cover the whole area of interest, but does not have to coincide with the help grid shown in Fig. 5. In the file "zdata" row 1, column 1 defines the bottom elevation of the lower left corner of the area, and the last row, last column gives the elevation of the upper right corner of the area, Fig. 11. Bottom elevation should be given from a fixed reference level below the wetland.

In order to relate the coordinates of "zdata" to the help grid, see section 4.1, it is necessary to define one coordinate and the step length in "zdata". This is done in the file "zkoord", which contains four values in a column. The first two rows are the x- respectively y-coordinate of the left lower node in the area described in "zdata", i e left upper value stated in "zdata". The third and fourth rows in "zkoord" state the step length in x- respectively y-direction between the nodes in "zdata". An example of how values stated in "zdata" and "zkoord" are interpreted is shown in Fig. 11.

When the routine "matsolve.m" is run, bottom elevation will be set to each node in the help grid using a four-point approximation of the values stated in "zdata". This is done in a help routine "elevation.m" called automatically by "matsolve.m".

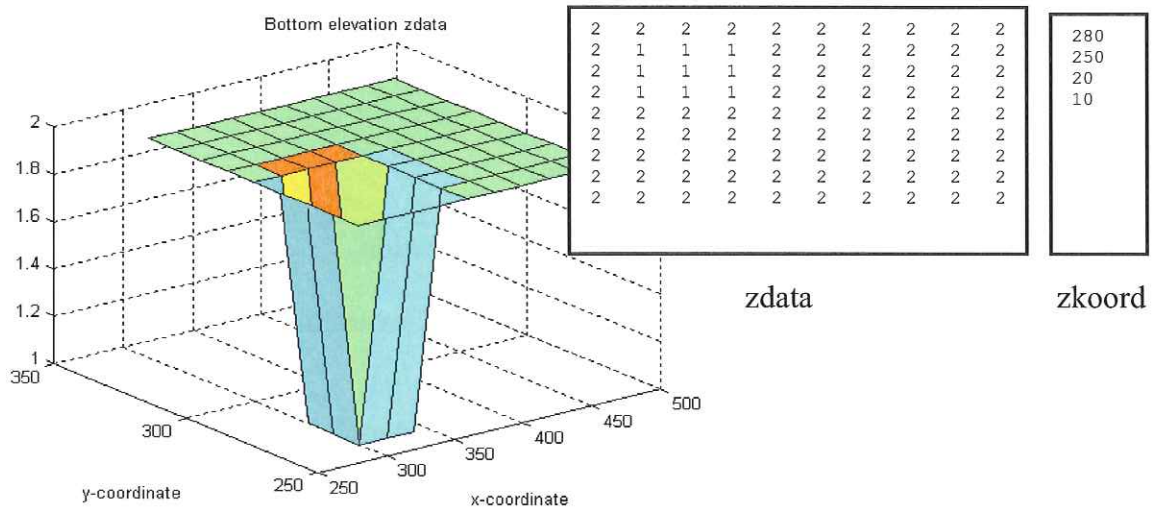

Fig. 11. Example of bottom elevation generated from the files "zdata" and "zkoord".

Elevation data can preferably be generated in GIS environment, but needs to be in raster format. The generated raster file should be loaded into "elevation.m". "elevation.m" is automatically run from "matsolve.m" and transforms the raster data to fit the coordinates of the help grid system. "zkoord" should be stated in the same manner as above also when using GIS to generate "zkoord".

4.4 Defining flow resistance: Files "kdata" and "kkoord"

In case of homogeneous flow resistance in the wetland, the F values can be stated directly in the code of "matsolve.m". In case of heterogeneous flow resistance the routine is similar to that of defining bottom elevation in 4.3. The inverse of flow resistance, conductivity, should then be stated in the input file "kdata" in the same way as bottom elevation is stated in "zdata". Coordinates if "kdata" should be stated in "kkoord", similarily to the way "zkoord" is created. The "kdata" file can also preferably be generated in GIS environment, and also needs to be in raster format. A help function to "matsolve.m", "Conductivity.m", converts "kdata" to conductivity values in each node of the help grid system. This function is automatically called by "matsolve.m" followed by conversion of conductivity values to flow resistance, F values.

4.5 Assembling of equation system and solution algorithm: File "matsolve.m"

Before the routine "matsolve.m" is run, you need to state directly in the code the following variables:

nx-value: This value is stated explicitly in "matsolve.m".

"matsolve.m" is based on the solution to Eqn. (10) considered in the form of Kmat*h + Cmat*z = RV. Kmat and Cmat are coefficient matrixes containing α-values of Eqn. (10). Vector RV contains zeros or boundary conditions. Because water depth h is also present in the α-values ($\alpha = h^{1+nx}/F$), the system is solved iteratively.

The solution can be expressed by h = (Kmat\(RV-Cmat*Zvector)), and repeting this with the new h- values used in the α-values until satisfactory accuracy is achieved.

"matsolve.m" is based on the solution to Eqn. (10) considered in the form of $A\, h^{(2+nx)} = B$, where A is a coefficient matrix containing the F-values of Eqn. (10) and statements related to boundary condition, h is the solution vector (hydraulic potential raised to the power of (2+nx)) and the vector B contains zero or boundary fluxes.

The solution can be expressed by a direct solution algorithm in the form $h = A^{-1}\, B^{1/(2+nx)}$.

The construction of Kmat, Cmat and RV is based on the coding in "code" and the numbering in "Numb". Which elements on the row in A that should be assigned a numerical value apart from zero depends on the number of neighbouring nodes. For instance, if the node (10,13) is numbered as 12, the equation for this node should be placed on row 12 in the equation system. If the surrounding nodes are 1, 11, 13 and 25, there will be contribution on row 12 in the columns 1, 11, 12, 13 and 25.

When "code" contains a number which is not zero or one, special forms of Eqn. (10) are considered as described in section 2.2.

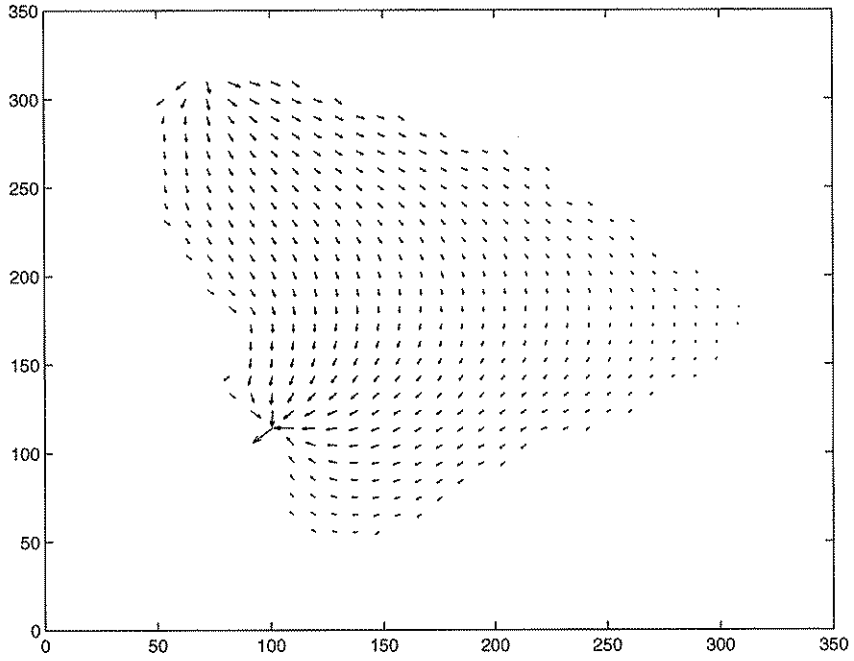

Fig. 12 Velocity vectors as calculated by the routine "velocities.m".

The routine concludes by plotting the surface- potential as seen in Fig. 9 as well as the equipotential lines as shown in Fig. 10. The surface- potential is stored in the matrix "pot" that corresponds to the coordinates of XX and YY.

4.6 Treatment of calculation results: Files "velocities.m" and "partrack.m"

The routine "velocities.m" calculates the velocity components in each grid node and plots the velocity vectors. To provide a suitable plot of the vectors, they are scaled as

$$\mathbf{V} = [|\mathbf{V}|^{nn}/|\mathbf{V}|]\, \mathbf{V} \qquad (16)$$

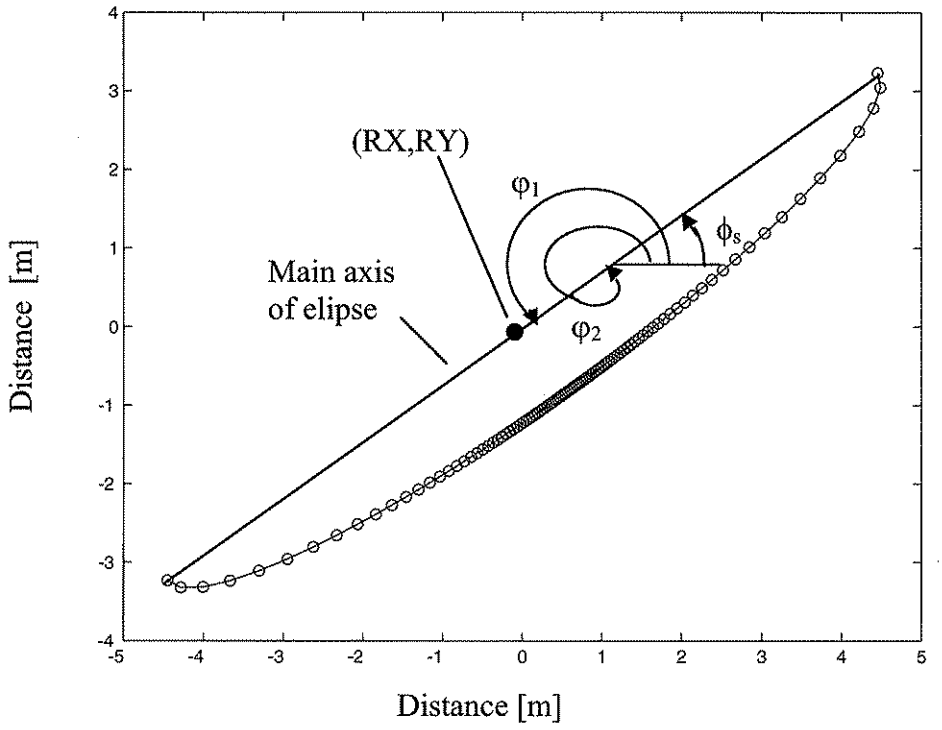

Fig. 13 Definition sketch of the release curve used for particle tracking. Circles denote release points.

The power nn is stated directly in "velocities.m".

The routine partrack releases a number, "particles" (set in "patrtrack.m), around the point (RX,RV). This point can be defined in different ways, but one possibility is that an inflow point defined in the routine "BCpoints.m" is used as a basis. The point (RX,RY) can, thus, be assigned values in "BCpoints". The release curve is defined in a segment of an elipse defined by the following parameters (cf. Fig. 13):

ϕ_s: Defines the angle of the main axis of the elipse from the horizon.

φ_1: Start angle of the release segment of the elipse.

φ_2: Stop angle of the release segment of the elipse.

aa = Length of main axis

bb = Length of shortest axis

The advantage of an elipse is that it can enclose a point (inflow) and also approximate a release along a line segment of the boundary.

The first angle in the list above is defined directly in "partrack.m", and in the current version, it is taken as φ_1 (but not more than 90°). The other four properties are defined in the indata file Relinf: first row contains φ_1 and φ_2 and the second row contains aa and bb (i.e. a 2x2 matrix). By means of the programme "checkrelease.m" it is possible to see how the particles are released before executing "partrack.m".

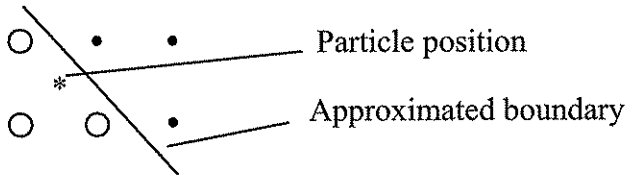

Fig. 14 Example of particle position (star) in relation to approximated boundary and calculation nodes (circles). The help grid includes both calculation nodes and "passive" nodes (points) and the particle can move freely inside the help grid.

The file "partrack.m" performs an iterative tracking of particles from their release points until the end-point value of the velocity in each movement satisfied the applied value with an accuracy of "eps" (assigned a value in "partrack.m"). The time step is set in "partrack.m" is depending on velocities and can be change by tuning the factor "DTfactor". The velocity vector in each point is deduced using a four-point approximation in the routine "hast_3".

After each time step, the programme analyses by means of the routine "domain1_2.m" if the current particle position is outside the approximated domain (cf. Fig. 4). This means that, in some cases, the particle can be in a position of the calculation domain where not all node values have assigned velocities, but still be inside the approximated domain (cf. Fig. 14). In this case, the four-point approximation will still assign a velocity vector for the particle position.

When the particle comes outside the approximated domain, the tracking is stopped and the travel time or residence time that the particle has spent in the domain in stored in "restime(1:particles)". Further, the programme analyses across which boundary segment the particle passed using the routine "outline_2.m". The enumeration convention adopted in the indata file "koord" and described in section 4.2. The number of the crossed boundary segment is stored in "mark(1:particles)". If the particle is released outside the approximated domain, which is possible with respect to the arguments above and Fig. 14, the tracking is not stopped. The governing criterion for stopping the tracking is that the particle passes a defined part of the boundary and remains outside after the last movement. This part of the boundary is defined between to boundary segments "segmentout1" and "segmentout2" in "partrack.m". If particles cross the boundary elsewhere from the defined part they are bounced back inside the boundary, using the routine "bouncing_2.m".

4.7 Dispersion and exchange with stagnant zones

Dispersion and exchange with stagnant zones, such as bed sediments, are treated in the particle tracking module. Both processes represent mixing or exchange between different zones of advection velocities, but the basic mathematical formulation differs. Dispersion is simulated as fickian diffusion where water parcels are moved within the velocity field. Exchange with stagnant zones is simulated explicit as a temporal immobilization of water parcels (Hays, 1966). Fickian diffusion is generally a good representation of exchange on a long time scale, a high Damköhler number. The mixing in wetlands occurs on several time scales and can be represented by either the dispersion mechanism, the water exchange with stagnant zones or both mechanisms together.

Dispersion is modelled using a random walk model, which is useful to describe various mixing mechanisms acting on solute tracers in the environment (Fischer, 1979). The positions of particles moving randomly in 1D can be shown to be Gaussian and their variance is related to the step size (Δx), time step (Δt) and elapsed time (t) as

$$\sigma^2 = \frac{t(\Delta x)^2}{\Delta t} \tag{16}$$

Solutes spreading according to Fickian diffusion also form a Gaussian (spatial) distribution with a variance as (Fischer, 1979)

$$\sigma^2 = 2Kt \tag{17}$$

where K is the longitudinal dispersion coefficient. Further, from fundamental theory of shear dispersion, the dispersion coefficient is proportional to squared velocity and a squared length characteristic to the flow conduit.(Fischer, 1979) In aquatic environments the characteristic length is taken as the water depth (h), rather than wetland length (Kadlec, 1994) In particular, Kadlec (1994) summarized dimensionless depth dispersion numbers K/Vh for different wetlands, where the number is taken as a constant $\beta = K/Vh$. A constant β is a slight contradiction versus the theory of shear dispersion. The motivation for the apparent contradiction stems from the presence of several mixing mechanisms in the water in addition to shear dispersion, and the complicated geometry of natural wetlands. Introducing $\beta=K/Vh$ in (6) and (7) yields

$$\Delta x = (2\beta Vh\Delta t)^{1/2} \tag{18}$$

Equation 8 is used in the model to determine the length of the dispersion step for each time step. The direction is taken as random in the horizontal plane. Because of the two dimensional modelling of dispersion, values of β used here are not to compare with values where 1D models have been used. The magnitude of dispersion is set by the factor "betha" in "dispersion.m". Betha=0 means no dispersion is present. The betha parameter is set in the code of "dispersion.m", which is a help function to "partrack.m"

Exchange of water with stagnant zones can partly be represented by heterogeneity in flow resistance, F-values. However, to evaluate other types of immobilization of water parcels in stagnant zones and on a smaller length scale an extra routine is formulated that does not depend on spatial resolution. Such stagnant zones are assumed to represent exchange of water with sediments and on spatial scales characteristic to the size of vegetation patches (several plants together) or stem diameter. Another special type of immobilization occurs due to stratification of floating vegetation that leads to stagnant zones at the water surface and underflows with mobile solutes (Harvey et al., 2005).

According to the model formulation, a water particle can be immobilized in the sediments for some time also when travelling in a fast flow path. The water exchange with stagnant zones is simulated as a random immobilization/mobilization of each particle in each time step. Assuming a first order transfer, the mass rate of change of the compound during one time can be formulated as

$$\frac{\Delta C}{C_0} = e^{\lambda \Delta t} \tag{19}$$

where ΔC=change of concentration of the compound, C_0=concentration of the compound at the beginning of the time step, λ=rate coefficient [s^{-1}]. The ratio $\Delta C/C_0$ can also be interpreted as the probability of transfer during the time step, which allows us to formulate the probability of a particle being immobilized when mobile, or mobilized if immobile, as

$$P(immobilized) = \left(1 - e^{-\lambda_1 \Delta t}\right) \tag{20}$$

$$P(mobilized) = \left(1 - e^{-\lambda_2 \Delta t}\right) \tag{21}$$

where λ_1=rate coefficient of immobilization [s^{-1}] and λ_2=rate coefficient of mobilization [s^{-1}]. The magnitude of exchange with stagnant zones is set by the parameters "λ_1" and "λ_2" in "partrack_m". Setting these to 0 means no exchange with stagnant zones. These parameters are set directly in the code of "partrack.m".

The "partrack.m" routine plots the particle trajectories as shown in Fig. 15 and uses Matlab´s "hist"-function to derive a simple residence time distribution. The following variables classify the distribution:

freq = number of particles within interval
pos = centered time position corresponding to freq
Int = interval width (equal)

After "partrack.m" is run, "PDFs.m" can be executed to calculate probability density functions (unity when integrated) that can be used in convolution procedures. The result is stored using "storage.m" and the convolution is performed using "convolute.m". The convolution simulates the residence time distributions in two subsequent wetlands in which an upper wetland feeds into a lower wetland. The method requires that two residence time probability density functions are available.

Because of numerical issues some particles may show unrealistic residence times. Those biased travel times may be recognized and omitted by running the routine "restime_modify.m" After this "Create_PDF.m" can be run to illustrate the residence time distribution.

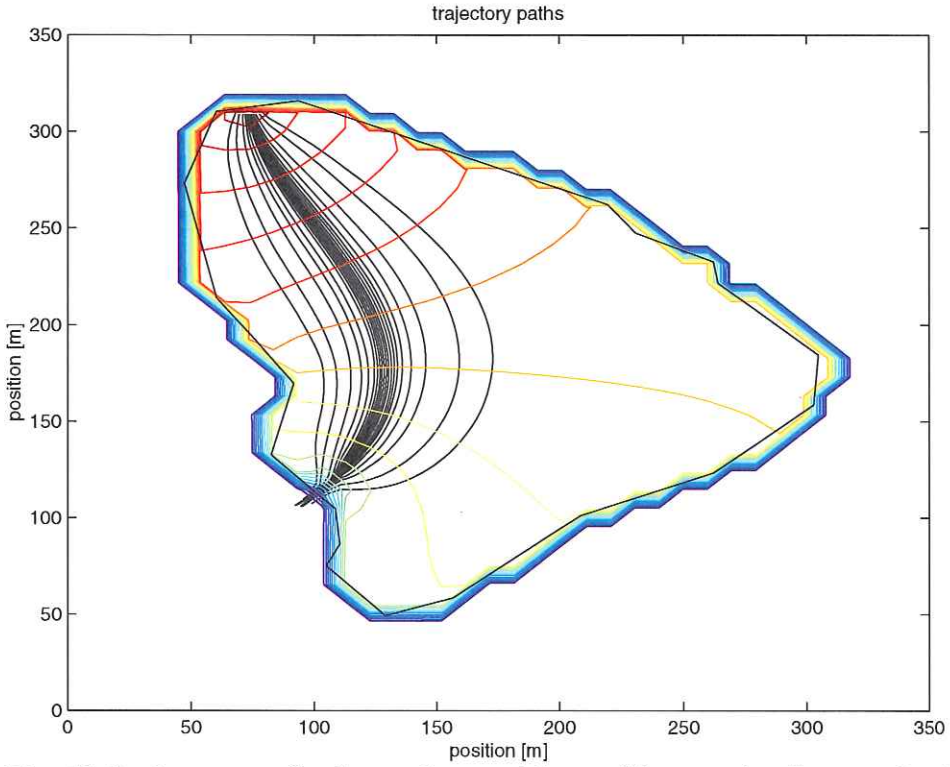

Fig. 15 Trajectory paths due to the particle tracking routine "partrack.m".

5. REFERENCES

Chow, Ven Te, 1959. Open-channel hydraulics McGraw-Hill Book Company, 1959. ISBN 07-010776-9.

Fischer H. B. et al, (1979), *Mixing in inland and coastal waters*, Academic press inc, London.

Harvey W. H., E. S. Saiers, J. T. Newlin., (2005), *Solute transport and storage mechanisms in wetlands of the Everglades, south Florida*, Water resources research, vol 41, W05009.

Hays, J.R., 1966. "Mass transport phenomena in open channel flow", PhD thesis, Dept. of Chemical Engineering, Vanderbilt University, Nashville, Tennessee, U.S.A.

Kadlec R. H., (1994), *Detention and mixing in free water wetlands*, Ecological engineering, 3(1994), 345-380.

Wörman, A., 2002. " Low-Velocity Flows in Constructed Wetlands: Physico-Mathematical Model and Computer Codes in Matlab-Environment". Swedish University of Agricultural Sciences, Dept. Of Biomertry and Informatics, report 76, ISSN 1650-1446

APPENDIX 1: FILES REQUIRED

MATLAB .m files

File aligned with left-hand margin is the main programme. These are listed in the order they should be executed. Indented files are used as sub-programmes by the main programme.

flownet.m
- **selline.m**
- **boundline.m**
- **plot1.m**

BCpoints.m
matsolve.m
- **assemble_1_4.m**
- **assemble_2_4.m**
- **Conductivity.m**
- **elevation.m**

velocities.m
partrack.m
- **release.m**
- **checkrelease.m**
- **domain1_2.m**
- **hast_3.m**
- **outline_2.m**
- **boundarypartrack_3.m**
- **bouncing_2.m**
- **depthapproximation.m**
- **dispersion.m**

Restime_modify
Create_PDF
PDFs.m
- **limitation.m**

Indata files

BC1
BC2x
BC2y
BC3
Koord
Relinf
Zdata
Zkoord
Kdata
Kkoord

TIDIGARE PUBLIKATIONER

2003-07-01 skedde en sammanslagning av Institutionen för biometri och informatik och Institutionen för lantbruksteknik.

Biometri och teknik

Examensarbeten

2005:01 Hårsmar, D. Bättre enskilda avlopp i Sigtuna kommun – möjligheter för bebyggelse i Odensala socken.

2005:02 Svensson, M. Desalination and the environment: Options and considerations for brine disposal in inland and coastal locations.

2005:03 Jakobsson, D. Retention av tungmetaller I en anlagd våtmark: studier av Vattenparken I Enköpings kommun.

2005:04 Leonardsson, J. & Östensson, E. Inverkan av torrsubstanshalt och temperatur på kompostens syrabildning.

2005:05 Ulff, D. Miljöpåverkansbedömning vid tillverkning av etanol från cellulosabaserade råvaror: ekologisk gård självförsörjande med drivmedel.

2004:01 Ericsson, N. Uthållig sanitet i Peru – En förstudie i staden Picota.

2004:02 Ekvall, C. LCA av dricksvattendesinfektion – en jämförelse av klor och UV-ljus.

2004:03 Wertsberg, K. Behandling av lakvatten med kemiska oxidationsmedel för att delvis bryta ned oönskade organiska föreningar – En studie utförd vid Hovgårdens avfallsanläggning i Uppsala.

2004:04 Degaart, S. Humanurin till åkermark och grönytor: avsättning och organisation i Göteborgsområdet.

2004:05 Westlin, H. Utvärdering av ett silotorksystem för spannmål utrustat med omrörare.

Rapport – miljö, teknik och lantbruk

2005:01 Jönsson, H., Vinnerås, B. & Ericsson, N. Källsorterande toaletter. Brukarnas erfarenheter, problem och lösningar.

2005:02 Gebresenbet, G. Effect of transporttime on cattle welfare and meat quality.

2005:03 de Toro, A. & Rosenqvist, H. Maskinsamverkan – tre fallstudier.

2005:04 Vinnerås, B. Hygienisering av klosettvatten för säker växtnäringsåterförsel till livsmedelsproduktionen.

2005:05 Tidåker, P. Wastewater management integrated with farming. An environmental systems analysis of the model city Surahammar.

2005:06 Sundberg, C. Increased aeration for improved large-scale composting of low-pH biowaste.

2005:07 Bernesson, S. Halm som energikälla.

2004:01 Bernesson, S. Life cycle assessment of rapeseed oil, rape methyl ester and ethanol as fuels – A comparison between large- and smallscale production.

2004:02 Elmquist, H. Decision-Making and Environmental Impacts.

Rapport – biometri

2004:01 Gustafsson, L. Tools for Statistical Handling of Poisson Simulation: Documentation of StocRes and ParmEst

Licentiatavhandling

02 Sundberg, C. 2004. Food waste composting – effects of heat, acids and size.

03 Forkman, J. Coefficients of variation: an approximate F-test.

Kompendium

2005:01 Lövgren, M. Publicering 2000-2004.

2004:01 Lövgren, M. Publicering 2000-2003.

2006:01. Low-Velocity Flows in Constructed Wetlands: Physico-Mathematical Model and Computer Codes in Matlab-Environment