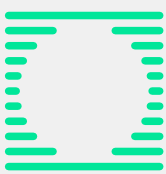
IBJplume

Eine Implementierung des Ausbreitungsmodells

VDI 3782 Blatt 1: Umweltmeteorologie — Atmosphärische Ausbreitungsmodelle —Gaußsches Fahnenmodell

Ing.-Büro Janicke, Dunum, August 2001



Ingenieurbüro Janicke
D-26427 Dunum
Alter Postweg 21

Gesellschaft für Umweltphysik
eMail: info@janicke.de
Internet: www.janicke.de

Vorbemerkung

Die Urheberrechte des Computerprogramms `IBJplume` liegen beim Ingenieurbüro Janicke, Dunum. Das Programm einschließlich des Quelltextes ist unter der GNU PUBLIC LICENCE verfügbar, die mit dem Programm verteilt wird. Das bedeutet im wesentlichen, daß es frei kopiert und weitergegeben werden darf, sofern dies kostenlos und wieder unter der GNU PUBLIC LICENCE erfolgt. Falls das Programm modifiziert und weitergegeben wird, ist der Programmname zu ändern, insbesondere darf der neue Name nicht mit „`IBJ`“ beginnen.

Diese Dokumentation, das Programm und Beispielrechnungen sind im Internet unter `http://www.janicke.de` verfügbar. Dort stehen auch Anmerkungen, Korrekturen und Beiträge Dritter.

1 Einleitung

Die Richtlinie VDI 3782 Blatt 1 beschreibt ein Gaußsches Fahnenmodell zur Berechnung der Ausbreitung von Spurenstoffen in der Atmosphäre. *IBJplume* ist die Modell-Implementierung, mit der die Arbeitsgruppe des VDI gearbeitet und die in der Richtlinie aufgeführten Beispiele durchgerechnet hat.

Abschnitt 2 beschreibt die Eingabeparameter für das Programm. Ergänzend ist in Abschnitt A die Struktur der verwendeten Ausgabedateien dargestellt.

2 Das Programm

Das Modell ist als Computer-Programm in ANSI-C realisiert. Es ist ein reines Text-Programm, läuft also unter UNIX in einem Terminalfenster, unter Windows in einem DOS-Fenster.

Das Programm ist Teil einer Reihe von Programmen zu Problemen der Stoffausbreitung in der Atmosphäre, die alle eine ähnliche Struktur besitzen, teilweise gleiche Module enthalten und das gleiche Datenformat verwenden. Sie unterliegen alle der GNU PUBLIC LICENCE, sind also kostenlos und werden im Quelltext weitergegeben. Die Urheberrechte liegen beim Ingenieurbüro Janicke, Dunum. Das Programm ist sorgfältig und nach bestem Fachwissen erstellt, es werden aber keinerlei Garantien für die Korrektheit der Resultate gegeben. Wer das Programm verwendet, hat sich selbst anhand der Beispiele und des Quelltextes davon zu überzeugen, daß es für seine Zwecke geeignet ist.

2.1 Benutzeranleitung

Das Programm arbeitet nicht interaktiv sondern verwendet eine Befehlsdatei, in der alle Aufgaben spezifiziert sind, die das Programm in einem Rechenlauf durchführen soll. Während der Rechnung schreibt das Programm ein Arbeitsprotokoll in die Protokolldatei und erstellt eventuell weitere Ausgabedateien. Alle diese Dateien haben vorgegebene

Namen und befinden sich in einem einzigen Verzeichnis, dem Arbeitsverzeichnis. Zum Aufruf des Programms ist einzugeben:

`IBJplume` *Arbeitsverzeichnis Option ...*

Mögliche Optionen sind:

`-e` :
Alle in der Befehlsdatei eingelesenen Textzeilen werden in der Protokolldatei protokolliert (*echo*).

`-h` :
Alle Aufgaben und Eingabeparameter werden am Bildschirm aufgelistet und anschließend das Programm beendet.

`-i`*Command* :
Die Datei mit dem Namen *Command* soll als Befehlsdatei verwendet werden (der Standardname ist `plmcmd.txt`).

`-l`*Log* :
Die Datei mit dem Namen *Log* soll als Protokolldatei verwendet werden (der Standardname ist `plmlog.txt`).

`-q` :
Es werden keine Meldungen auf den Bildschirm geschrieben (*quiet*).

`-v`*Verbose* :
Mit *Verbose* wird angegeben, wie ausführlich das Programm Meldungen auf dem Bildschirm und in der Protokolldatei ausgibt. Der Standardwert ist 0 (keine Ausgabe).

Die Befehlsdatei (Standardname ist `plmcmd.txt`) ist eine reine Textdatei, kann also vom Benutzer mit einem Text-Editor erstellt werden. Die Protokolldatei (Standardname ist `plmlog.txt`) ist ebenfalls eine Textdatei, die vom Programm bei jedem Programmlauf angelegt wird. Die Rechenergebnisse stehen entweder in der Protokolldatei oder in separaten Ausgabedateien.

Die Befehlsdatei besteht aus mehreren Abschnitten. Jeder Abschnitt definiert eine Aufgabe mit den dazugehörigen Parametern. Eine vollständige Auflistung aller möglicher Aufgaben und der zugeordneten Parameter wird in Abschnitt 2.2 gegeben.

Die erste Zeile eines Abschnittes beginnt mit dem Zeichen „`*`“, unmittelbar gefolgt von einem Namen, und definiert die Teilaufgabe, die ausgeführt werden soll, beispielsweise

```
*Dimensionierung
```

Signifikant ist hierbei nur der erste Buchstabe, also das „`D`“, wobei auch nicht zwischen Klein- und Großschreibung unterschieden wird. Genauso gut hätte man also auch schreiben können

```
*dimensions
```

Es folgen in diesem Falle die Parameter, die zur Dimensionierung der Felder benötigt werden, also z.B.

```
*Dimensionierung    ' Beispiel
nh    3             ' Ausbreitungsparametersätze für 3 Quellhöhen
nw    1             ' Eine Windrichtung
np    6             ' Definition von 6 Monitorpunkten
```

Bei den Parameternamen sind nur die ersten beiden Zeichen signifikant und es wird ebenfalls nicht zwischen Klein- und Großschreibung unterschieden. Anschließend folgen, durch Leerzeichen und/oder Tabulatoren getrennt, der oder die Werte. Dabei kann es sich um ganze Zahlen, Gleitkommazahlen oder Zeichenketten handeln. Zeichenketten können in Hochkommata eingeschlossen sein. Auch die Zeile, welche die Definition der Aufgabe enthält, kann nach dem Aufgabenamen noch Parameterwerte enthalten.

Ein Apostroph leitet einen Kommentar ein, der beim Einlesen überschlagen wird. Beginnt eine Zeile mit einem Leerzeichen oder einem Minuszeichen, wird die ganze Zeile als Kommentar angesehen.

Nachdem die erste Aufgabe eingelesen ist, wird die Dimensionierung der internen Felder vorgenommen, die später nicht mehr geändert werden kann. Die erste Aufgabe muß also immer `*D` sein. Auch für Eingabeparameter, für die mehr als ein Wert anzugeben ist, wird der Speicherplatz in diesem Schritt angelegt. Wenn die Anzahl der Werte vom Benutzer festgelegt werden kann, existiert immer ein Parameter im Abschnitt `*D`, der für die Dimensionierung verwendet wird. Beispielsweise kann der Benutzer festlegen, an wievielen Aufpunkten die Konzentration ausgewiesen werden soll:

```
*Dimensionierung    ' Beispiel
np   6              ' 6 Aufpunkte
*Aufpunkte          ' Festlegung der Aufpunkte
xp   100 200  500 1000  2000   5000  ' x-Koordinate (m)
yp   0     0    0    0     0      0  ' y-Koordinate (m)
```

Die Namen der Parameter sind über alle Abschnitte hinweg eindeutig und dürfen nur in dem Abschnitt gesetzt werden, für den sie vereinbart sind. Sie behalten ihren Wert solange bei, bis er explizit geändert wird.

2.2 Eingabeparameter

Das Programm *IBJplume* erkennt folgende Aufgaben:

`*C`
: Es werden alle Werte auf dem Gitter und den Aufpunkten gelöscht.

`*D`
: Die Dimensionen aller verwendeten Felder werden festgelegt. Diese Aufgabe darf nur einmal definiert werden und muß als erste in der Befehlsdatei erscheinen.

`*A`
: Eien Reihe von Aufpunkten wird definiert, an denen die bodennahe Konzentration, trockene oder nasse Deposition oder umgewandelte Konzentration berechnet und in die Protokolldatei ausgerschrieben wird.

`*G`

Ein Gitter von Aufpunkten wird definiert, für das die berechneten Werte in eine eigene Datei ausgeschrieben werden.

`*P`

Die für die Ausbreitungsrechnung benötigten Parameter (Meteorologie, Ausbreitungsparameter) werden festgelegt. Diese Aufgabe kann beliebig innerhalb einer Rechnung auftreten und bewirkt jedesmal eine Neuberechnung der internen Felder.

`*Q`

Die Quelldaten (Position, Ausdehnung, Quellstärke) werden festgelegt. Diese Aufgabe kann beliebig innerhalb einer Rechnung auftreten.

`*L plm|pnt`

Tabellierung der aktuellen Fahnenwerte in der Protokolldatei für alle Quellen. In der Protokolldateien werden die folgenden Parameter ausgeschrieben:

Bei `plm` Auflistung der Fahnenfunktionen:

`Iy`	Normierte Konzentration über y integriert
`Iyz`	Normierte Konzentration über y und z integriert
`Umwandl.`	Umgewandelter Anteil der Konzentration
`Sigma-Y`	Horizontale Fahnenbreite
`Konzentr.`	Konzentration am Boden auf der Fahnenachse

Die normierte Konzentration ist auf die Quellstärke 1 ME/s bezogen.

Bei `plm` Auflistung der Werte an den Aufpunkten:

`Cm`	Konzentration
`Fd`	Massenstromdichte der trockenen Deposition
`Fw`	Massenstromdichte der nassen Deposition
`Tr`	Umgewandelte Konzentration

`*S cnc|dry|wet|trn|qtl`

Ergebnistabellen werden ausgeschrieben. Es können mehrere Dateien mit einem Schreibbefehl ausgeschrieben werden, z.B. mit „`*S cnc+dry`“. Die Ergebnistabellen sind Mittelwerte über alle seit dem letzten `*C` abgearbeiteten Intervalle.

`*T *|`*Abschnittbuchstabe*

Ausgabe der aktuellen Parameter in die Protokolldatei. Mit `*` werden alle Parameterabschnitte, sonst nur der spezifizierte Abschnitt ausgegeben.

`*W` *aks*

Die Ausbreitungsklassenstatistik *aks* (DWD-Format) wird eingelesen und automatisch abgearbeitet.

`*E`

Das Programm wird beendet.

Die einzelnen Eingabeparameter sind diesen Aufgaben zugeordnet, dürfen also nur in dem betreffenden Abschnitt gesetzt werden. Sie stellen entweder ganze Zahlen, Gleitkommazahlen oder Zeichenketten dar und werden zu Beginn der Rechnung mit einem

Standardwert belegt. Sie behalten ihren Wert solange bei, bis er durch Eingabe eines neuen Wertes geändert wird. Ausnahmen hiervon sind explizit aufgeführt. Alle Längen sind in Metern anzugeben, alle Zeiten in Sekunden.

Die einem Parameter zugewiesenen Werte folgen dem Namen, von diesem und untereinander durch Leerzeichen und/oder Tabulatoren getrennt.

In der folgenden Tabelle steht für jeden Parameter in einer Zeile der *Name*, anschließend der *Datentyp* (*integer*, *float* oder *string*), dahinter in Klammern die *Anzahl* der Werte und schließlich die *Standardsetzung*. In den darauf folgenden Zeilen ist die Bedeutung und Handhabung dieses Parameters beschrieben.

`*C` Clear

`*D` Dimensionierung

`mc` *integer*(1) `4000`
Maximale Anzahl von Bytes, die in einer Eingabezeile vorkommen können.

`me` *string*(1) `g`
Masseneinheit (wird nicht weiter ausgewertet).

`nh` *integer*(1) `1`
Anzahl n_h der Definitionshöhen für die Ausbreitungsparameter.

`np` *integer*(1) `0`
Anzahl n_p der Aufpunkte.

`nq` *integer*(1) `1`
Anzahl n_q der Quellen.

`nw` *integer*(1) `36`
Anzahl n_w der Windirchtungen.

`op` *string*(1) `""`
Optionen zur Durchführung der Rechnung. Die bisher einzige Option ist:

`taluft` Ausbreitungsrechnung nach der alten TA Luft (1986). Hierbei wird für Quellhöhen über 100 m ein anderer Satz von Ausbreitungsparametern gewählt, die Exponenten der Windgeschwindigkeit werden geändert, die Mischungsschichthöhe wird auf unendlich gesetzt und als repräsentative Transportgeschwindigkeit $\overline{u}$ wird die Windgeschwindigkeit in effektiver Quellhöhe verwendet.

`ti` *string*(1) `TEST`
Bezeichnung (*title*) für den Rechenlauf für Dokumentationszwecke.

`zp` *float*(1) `1.5`
Z-Koordinate der Aufpunkte und Gitterpunkte (m).

`*A` Aufpunkte

`xp` *float*(n_p) `0`
X-Koordinate der Aufpunkte (m).

`yp` *float*(n_p) `0`
Y-Koordinate der Aufpunkte (m).

`*G` Gitter

`dd` *float*(1) `0`
Maschenweite (m).

`nx` *int*(1) `0`
Anzahl der Gitterpunkte in x-Richtung.

`ny` *int*(1) `0`
Anzahl der Gitterpunkte in y-Richtung.

`x0` *float*(1) `0`
X-Koordinate des linken Randes (m).

`y0` *float*(1) `0`
Y-Koordinate des unteren Randes (m).

`*P` Parameter

`nu` *int*(1) `5`
Anzahl der Untersektoren für eine Windrichtung.

`dw` *float*(n_w) `1`
Häufigkeit einer Windrichtung.

`re` *float*(1) `10`
Richtung des ersten Windrichtungssektors (Grad gegen Nord im Uhrzeigersinn).

`kl` *string*(1) `III/1`
Klug/Manier-Klasse. Bei Angabe von `kl` werden die Parameter `hm`, `ew`, `py`, `qy`, `pz`, `qz` automatisch bestimmt, es sei denn, sie werden explizit vorgegeben.

`ha` *float*(1) `10`
Höhe des Anemometers z_A über Grund (m).

`ua` *float*(1) `3.0`
Windgeschwindigkeit in Anemometerhöhe u_A (m/s).

`l0` *float*(1) `1000.0`
Charakeristische Länge L_0 bei Flächenquellen (m).

`a0` *float*(1) `50.0`
Maximale Anzahl a_0 von Teilquellen.

`rh` *int*(1) `0`
Index der Bodenrauhigkeit (0: mittel, 1: hoch) zur Bestimmung des Exponenten der Windgeschwindigkeit.

`vd` *float*(1) `0`
Depositionsgeschwindigkeit v_d (m/s).

`vs` *float*(1) `0`
Sedimentationsgeschwindigkeit v_s (m/s).

`ar` *float*(1) `0`
Auswaschrate Λ (1/s).

`ur` *float*(1) `0`
Umwandlungsrate k (1/s).

`ew` *float*(1) `0.31`
Exponent der Windgeschwindigkeit m.

`hm` *float*(1) `800.0`
Mischungsschichthöhe h_m (m).

`hs` *int*(n_h) `100.0`
Definitionshöhen für die Ausbreitungsparameter (m).

`py` *float*(n_h) `0.504`
Ausbreitungsparameter F.

`qy` *float*(n_h) `0.818`
Ausbreitungsparameter f.

`pz` *float*(n_h) `0.265`
Ausbreitungsparameter G.

`qz` *float*(n_h) `0.818`
Ausbreitungsparameter g.

`*Q` Quelle

`xq` *float*(n_q) `0`
X-Koordinate der Quelle (m).

`yq` *float*(n_q) `0`
Y-Koordinate der Quelle (m).

`hq` *float*(n_q) `100.0`
Bauhöhe der Quelle H (m).

`eq` *float*(n_q) `1.0`
Quellstärke Q_0 (ME/s).

`aq` *float*(n_q) `0`
Länge der Quelle in y-Richtung vor der Drehung (m).

`bq` *float*(n_q) `0`
Breite der Quelle in y-Richtung vor der Drehung (m).

`rq` *float*(n_q) `0`
Drehung der Quelle (Längsrichtung gegen Nord in Grad).

`uf` *float*(n_q) `0`
Endüberhöhung der Abgasfahne (m).

`wf` *float*(n_q) `0`
Weite des Fahnenanstiegs (m).

`ni` *int*(n_q) `100`
Anzahl der Intervalle zur Bestimmung der Fahnenfunktion.

`li` *float*(n_q) `30.0`
Länge eines Intervalls (m).

`*W` Wetterstatistik *aks*

`*L` Liste `[plm|pnt]`

`nd` *int*(1)`0`
Anzahl der beim Auflisten jeweils auszulassenden Punkte.

`*S` Speichern `[con|dry|wet|trn|qtl]`

`qt` *float*(1)`0.95`
Quantil, das bei Angabe der Option `qtl` berechnet und abgespeichert werden soll.

`rv` *float*(1)`0.0`
Referenzwert (wird nicht weiter ausgewertet).

`fa` *float*(1)`1.0`
Faktor, mit dem die Werte vor dem Abspeichern im Format `fo` multipliziert werden.

`fo` *string*(1)`%12.2e`
Darstellungsformat beim Abspeichern der Werte (C-Notation).

`fi` *string*(1)`""`
Dateiname, unter dem abgespeichert wird. Wenn leer, wird der Name der Option zur Bildung des Dateinamens verwendet.

`id` *string*(1)`""`
Identifikation (wird nicht weiter ausgewertet).

`mo` *string*(1)`text`
Modus zum Abspeichern der Daten (`text` oder `binary`).

`se` *string*(1)`j-:i+`
Index-Sequenz, in der die Werte abgespeichert werden. Der Index i bezeichnet hierbei die x-Richtung, der Index j die y-Richtung. Je weiter links ein Index in der Sequenz steht, desto langsamer läuft er in der Schleife zum Abspeichern, mit `j-:i+` werden die Werte also genordet ausgeschrieben.

A Dateistruktur von DMN-Dateien

Alle Dateien, die Ein- oder Ausgabefelder (Tabellen) repräsentieren, sind nach dem gleichen Prinzip aufgebaut. Sie enthalten zuerst einen Kopf, in dem alle Angaben zur Struktur und Darstellung der Tabelle stehen. Es folgt der Rumpf mit der eigentlichen Tabelle. Dieser Rumpf kann unmittelbar an den Kopf angehängt sein, wenn die Tabelle im Textformat (Standard) ausgeschrieben wird. Bei unformatierter Darstellung bildet der Rumpf immer eine eigene Datei.

Der Kopf ist eine Textdatei mit der Namenserweiterung `dmna`, die aus einzelnen Zeilen besteht. In jeder Zeile ist ein Parameter definiert. Der Name des Parameters steht zu Beginn der Zeile, gefolgt von einem oder mehreren Werten. Zulässige Trennungszeichen sind Leerzeichen, Tabulator und Semikolon, die einzeln oder kombiniert verwendet werden können. Die Zeile kann durch LF oder CR+LF abgeschlossen sein.

Neben den vom Programm benötigten Parametern kann der Kopf auch weitere Parameter enthalten. Parameter, die das Programm nicht kennt, werden ignoriert. Der Kopf endet mit einer Zeile, die zu Anfang einen Stern, also `*`, enthält. Mit der nächsten Zeile beginnt der Rumpf, sofern er mit dem Kopf zusammen eine einzige Datei bildet. Die (formatierten) Tabellenelemente im Rumpf werden durch Leerzeichen, Semikolon, Tabulator, CR oder LF getrennt. Die Tabelle wird durch eine Zeile, die mit drei Sternen beginnt, beendet.

Folgende Parameter im Kopf der Datei werden vom Programm erkannt und interpretiert (die Namen müssen klein geschrieben sein):

`data` *string*(1)
: Name der Datei, der die eigentliche Tabelle enthält. Ist `data` nicht spezifiziert oder hat es den Wert „`*`“, dann wird bei formatierter Ausgabe die Tabelle in die gleiche Datei geschrieben wie der Kopf. Bei unformatierter Ausgabe wird der Dateiname des Kopfes übernommen, er erhält aber statt „`.dmna`“ die Endung „`.dmnb`“. Falls in `data` eine Pfadangabe gemacht ist, gilt diese relativ zu dem Verzeichnis, in dem der Kopf gespeichert ist.

`dims` *integer*(1)
: Anzahl der Dimensionen (maximal 5).

`fact` *float*(1)
: Faktor, mit dem bei formatierter Ausgabe alle Datenelemente vom Typ *float* oder *double* multipliziert werden, bevor sie im angegebenen Format ausgeschrieben werden. Bei der Eingabe formatierter Daten werden diese Datenelemente nach dem Einlesen durch `fact` dividiert. Der Faktor wird bei Datenelementen, für die im Format `form` ein eigener Faktor definiert ist, ignoriert.

`form` *string*(1)
: Format, nach welchem bei formatierter Speicherung die Daten abgelegt sind. Bestehen die Tabellenelemente des gespeicherten Feldes aus mehreren Datenelementen (Datenstruktur), dann ist für jedes Datenelement eine Formatangabe erforderlich, und alle Einzelformate verkettet ergeben die Zeichenkette `form`.
$Format = Format_1 Format_2 \ldots$

$Format_i$ = *Name*`%(*`*Factor*`)`*Length*`.`*PrecisionSpecifier*

Es bedeuten:

Name	Name des Datenelementes (optional).
Factor	Skalierungsfaktor (optional einschl. Klammern).
Length	Länge des Datenfeldes.
Precision	Anzahl der Nachkommastellen (bei *float*-Zahlen).
Specifier	Umwandlungsangabe.

Der Skalierungsfaktor *Factor* wird genauso gehandhabt wie der Parameter `fact`. Die Längenangabe *Length* ist die Mindestlänge des Datenfeldes. Sie kann überschritten werden, wenn dies zur korrekten Darstellung der Zahl erforderlich ist. Zwischen den Zahlen steht immer mindestens ein Trennungszeichen.

Folgende Umwandlungsangaben sind möglich:

Spec.	Typ	Länge	Beschreibung
`c`	*character*	1	einzelne Buchstaben
`d`	*integer*	4	Dezimalzahl
`x`	*integer*	4	Hexadezimalzahl
`f`	*float*	4	Festkommazahl (ohne Exponent)
`e`	*float*	4	Gleitkommazahl (mit Exponent)

Den Angaben `f` und `e` kann ein `l` vorangestellt sein (*double* mit Länge 8 Bytes), den Angaben `d` und `x` ein `h` (*short integer* mit der Länge 2 Bytes).

Gleichartige Formatangaben können zusammengefaßt werden:
`vx%5.2fvy%5.2fvz%5.2f` ist äquivalent zu `vx%[3]5.2f`

`hghb` *integer*(`dims`)
Höchster Indexwert für die verschiedenen Laufindizes.

`lowb` *integer*(`dims`)
Niedrigster Indexwert für die verschiedenen Laufindizes.

`mode` *string*(1)
Bei `binary` sind die Daten unformatiert gespeichert, sonst formatiert.

`sequ` *string*(1)
Angabe, in welcher Indexfolge die Daten gespeichert sind. Normalerweise läuft der am weitesten rechts stehende Index am schnellsten (C-Konvention). Dies entspricht bei einem 3-dim. Feld A_{ijk} der Angabe `i+,j+,k+` (oder `i+:j+:k+`). FORTRAN speichert gemäß `k+,j+,i+`. Ein Minuszeichen statt des Pluszeichens bedeutet, daß der betreffende Index rückwärts läuft. Es können auch Teilbereiche ausgewählt werden: `j=10..1/1,i=5..25/1,k=1`. Die Angabe `/`n bedeutet, daß der betreffende Index des Ausschnittes mit dem Wert n anfängt. Wird mit `sequ` ein Auschnitt des Datenfeldes definiert, dann beziehen sich die Indexgrenzen `lowb` und `hghb` auf die ursprünglichen Indexdefinitionen.

`size` *integer*(1)
: Länge der Datenstruktur (*record size*) in Bytes. Bei formatierter Speicherung muß die aus der Formatangabe resultierende Summe der Längen der einzelnen Datenelemente gleich `size` sein.

Beispiel:

Ein Feld von Gleitkommazahlen $A_{ijk} = 100i + 10j + k, i = 1..3, j = 2..4, k = 0..1$ wird in horizontalen Schichten gespeichert:

```
form  %4.1f
mode  text
sequ  k+,j-,i+
fact  1.000e-001
dims  3
size  4
lowb  1 2 0
hghb  3 4 1
*
  14.0  24.0  34.0
  13.0  23.0  33.0
  12.0  22.0  32.0

  14.1  24.1  34.1
  13.1  23.1  33.1
  12.1  22.1  32.1

***
```