Gekko og Statistikbankens nye API

Thomas Thomsen, 6/11 2013

Pr. 1/11 2013 har der været adgang til Danmarks Statistiks Statistikbank via et offentligt tilgængeligt API (api.statbank.dk). Det gør det bl.a. muligt at downloade tidsserier på en transparent og nem måde, og det er tanken at udbygge Gekko med sådanne muligheder.

I det følgende gennemgås et forslag til en måde at lave interfacet mellem Gekko og Statistikbanken på. Kommentarer er meget velkomne: der er jo en del detaljer i det, hvilket fodnoteniveauet også afspejler!

Broen mellem Gekko og Statistikbanken foreslås lavet vha. forespørgsler i JSON-format, som returnerer PC-Axis-filer. JSON svarer lidt til XML, men er enklere, nemmere at læse og fylder mindre (der vendes i afsnit 4 tilbage til, hvordan disse filer kunne tænkes dannet). Det er det centrale inputformat mht. Statistikbankens API, mens der er mange flere muligheder mht. API'ets outputformat. Valget er faldet på Axis-filer som output, fordi disse er (a) komplette i den forstand at de har alle metadata med, (b) nemme at læse (for tidsserie-forespørgsler i hvert fald), og (c) velafprøvede igennem mange år, og der findes software til at læse/skrive/viewe dem.[1]

Statistikbanken består af et antal tabeller (tusindvis), svarende til SQL-tabeller med et bestemt antal kolonner. Nogle af kolonnerne er dimensioner ("codes"), og man kan filtrere på disse (ved at udvælge "values" fra kolonnerne). Eksempelvis kan en tabel indeholde en kolonne med koden/navnet "tid", svarende til tidsdimensionen. Her kunne man eksempelvis filtrere på værdierne 2010, 2011 og 2012 i sin forespørgsel (eller vælge alle tidsperioder med "*"). Vha. en JSON-forespørgsel kan man således udtrække fra en given tabel, og dette udtræk kan gemmes som en Axis-fil.

Ofte vil man ønske at foretage udtræk fra et antal forskellige tabeller, og dette vil nødvendigvis give et tilsvarende antal Axis-filer. For at holde tingene samlet foreslås det derfor, at JSON-filerne samles i én zip-fil, mens de resulterende Axis-filer ligeledes samles i én zip-fil. På den måde kunne man eksempelvis kalde sit forespørgselssystem for js_statbank.zip, og Gekko danner så en korresponderende fil ved navn px_statbank_YYYYMMDD.zip indeholdende Axis-filer svarende til forespørgslerne.[2] Placeringen af disse filer (altså hvilke mapper de ligger i) skal naturligvis kunne styres via nogle options. Der kunne her være en idé i også at gemme JSON-filerne i outputfilen (altså i den zip-fil som samler Axis-filerne). JSON-filerne vil typisk være ret små relativt til de data de producerer, og på den måde vil man altid senere kunne se, hvordan en given tidsserie eller observation er fremkommet. Man ville endda kunne bede en px_...-fil om at auto-opdatere sig selv (dvs. producere en ny fil ud fra samme forespørgsler, og give den samme navn bortset fra datoen).

[1] Eksempelvis findes der et interface til R (pakken pxR).

[2] "px" står for PC-Axis, og YYYYMMDD er en dato. Hvis der er udtrukket flere gange på en dag, kunne der tilføjes "a", "b", "c" osv. til YYMMDD. Altså at px_statbank_20131105.zip er den første fil den dag, px_statbank_20131105a.zip den anden osv. Hypotesen er, at udtræk typisk ikke foretages flere gange samme dag, så tilføjelse af klokkeslæt ville nok være overkill.

Der udvikles endvidere en Axis-læser til Gekko, som indlæser tidsserierne fra px_...-filen som en Gekko-databank i RAM.[3] Mht. tidsserienavne påtænkes der at følge en ret rigid navngivning, nemlig at hver af disse tidsserier får et navn genereret ud fra tabelnavnet og den valgte filtrering. Eksempelvis kunne navnet "PRIS6_VAREGR_011100_enhed_100" svare til et prisindeks for kødprodukter, idet navnet indikerer at der er valgt tabellen "PRIS6", filtreret på dimensionen "VAREGR" med værdien 011100, og filtrereret på dimensionen "enhed" med værdien 100 (hvilket betyder at tallet er opgjort som et indeks). I Gekko kan sådanne variabelnavne være vilkårligt lange, og et sådant langt navn vil give en unik identifikation af tidsserien. Man kunne så efterfølgende forestille sig en omdøbning til enklere og kortere navne, afhængigt af hvad tidsserien skal bruges til.[4] Men som udgangspunkt bevares dette lange navn for serien, og der dannes også en label til tidsserien ud fra metainformationerne i Axis-filen.[5] Der vil så efterfølgende kunne skrives en fil i Gekkos filformat for databanker (tsdx), eksempelvis med navnet statbank_YYYYMMDD.tsdx. Som for Axis-filerne (px_...) kunne man forestille sig, at tsdx-filen også indeholder forespørgslerne som JSON-filer. Tsdx-formatet er i forvejen en zip-fil indeholdende underfiler, så der er intet til hinder for at inkludere flere filer dér. På den måde ville en tsdx-fil med udtræk fra Statistikdatabanken også kunne auto-opdatere sig selv hvis det ønskes.

1. Inkrementale forespørgsler

Et givet system af forespørgsler (samlet i en js_...-fil) vil ofte returnere mange af de samme data som sidst, enten fordi tabellen som helhed ikke er blevet opdateret, eller fordi tabellen kun opdateres for visse perioder. For at reducere trækket på Danmarks Statistiks databaser samt reducere downloadtiden kunne der implementeres et system, som gør det muligt kun at udtrække ændrede tal.[6] Givet en eksisterende px_...-fil fra en given dato (og uændrede forespørgselsfiler) bliver spørgsmålet, så hvor meget af data som kan tages fra den eksisterende px_...-fil, og hvor meget der skal downloades pånyt. Det er relativt nemt at forespørge om en given tabel har været opdateret siden en bestemt dato. Lidt vanskeligere er det med tabellens opdaterede perioder, men DST/Lars Knudsen mener, at det ville være forholdsvist overkommeligt

[3] Der findes pt. ikke nogen open-source PC-Axis-læser til .NET. Til tidsserieformål er formatet dog så simpelt, at det kan indlæses ud fra meget få regler. Så planen er at lave en Axis-læser til Gekko, som kan ligge i public domain og evt. bruges af andre interesserede. Axis-læseren vil lave konsistenstjek af, at antallet af læste celler passer med de valgte dimensioner fra tabellerne mv., så indlæsningen burde kunne laves på ret så fejlsikker måde.

[4] Eksempelvis hvis tidsserierne skal viderebearbejdes i AREMOS via en tsd-fil, eller man kan have egne filer som omdøber fra det ene navn til det andet (eksempelvis hvis man har brug for at danne de ældre såkaldte S-koder svarende til tidsseriedatabanken DSTB). Statistikbanken har i mange år kunnet levere udtræk som en AREMOS-tsd-fil, men et af problemerne med dette er begrænsninger i tsd-formatet (længden på variabelnavne, og antallet af betydende cifre). Når der downloades som AREMOS-tsd-fil gives serierne derfor fortløbende navne (eksempelvis ville de hedde PRIS61, PRIS62, PRIS63, ... i den fil som returneres fra Statistikbanken), fordi der er en begrænsning på 15 karakterer i tsd-formatet. Dette kan senere give problemer med identifikationen, som jo ikke ud fra variabelnavnet kan kobles entydigt til tabel og tabelkoder. Det såkaldte asb-format (time series batch) er lidt mere fleksibelt, men er virkeligheden blot AREMOS SERIES-ordrer med de begrænsninger det giver.

[5] Denne label indeholder som udgangspunkt hele metainformationen. På sigt kunne det måske være en idé i Gekko at kunne operere med korte og lange labels, eller muligheden for i Gekko at have flere label-felter med metainformation. Da Gekko-labels kan være vilkårligt lange, kunne man også bare indlægge al information i denne label som en lang xml-tekststreng, som Gekko så kan vise på forskellige måder alt afhængigt af brugerens ønsker. Da de lange labels vil ligne hinanden, kunne forskellige former for komprimering komme på tale internt i Gekko.

[6] Dette svarer lidt til en inkremental backup, hvor man kun gemmer filer som har ændret sig siden forrige backup.

at udvide API'et, så man kan forespørge en tabel om hvorvidt data for en given periode (for eksempel 2013q3) har været ændret siden et bestemt tidspunkt (eksempelvis det tidspunkt den eksisterende px_...-fil blev lavet på).[7]

Ud fra en given px_...-fil (alternativt tsdx-fil) vil man således kunne spørge Statistikbanken om en given tabel har været opdateret siden filen blev dannet (og eventuelt hvilke perioder der er blevet opdateret). Så ville man kunne tilpasse forespørgslen til kun at downloade de relevante data, og data kunne merges med den eksisterende px_...-fil, så man får en ny komplet px_...-fil som gerne skulle være identisk med hvad man ville få ved et fuldt udtræk. Den nye px_...-fil bør i så fald indeholde en log, som fortæller hvilke data der er kopieret fra den ældre px_...-fil (og hvad filen hedder og evt. hvad dens hash/MD5 er). Denne ældre px_...-fil kan så være delvist dannet ud fra en endnu ældre px_...-fil osv. En gang imellem vil man måske foretrække at gøre rent bord med et fuldt download, men i princippet burde det inkrementale system kunne gøres ret så fejlsikkert.[8]

2. Fejl og validering

Der kan naturligvis opstår fejl, hvis der er problemer med servere eller internetforbindelse, men udover dette kan en forespørgsel også fejle, hvis JSON-filen ikke er valid. Hvis det drejer sig om manglende parenteser eller lign., ville Gekko kunne gøre opmærksom på dette (syntaksproblemer). Hvis det drejer sig om forkerte navne på tabeller eller dimensioner i disse, vil API'et fejle med en meddelelse om dette. Her kunne man godt lade Gekko hjælpe, evt. ved at kunne kalde API'et i en slags fejlsikret tilstand. I denne tilstand tjekkes først om tabellen overhovedet eksisterer, og om den evt. er ophørt. Derefter tjekkes dimensionerne for, om de overhovedet eksisterer (det svarer lidt til at tjekke, om SQL-tabellen findes, og om den har kolonnenavne svarende til de koder/filtre man ønsker at lægge på).

3. Automatisering og batch

Man kan kalde Gekko via parametre til gekko.exe, så det ville ikke være vanskeligt at have en bat-fil, som starter Gekko op med en given kommandofil (som går i gang med forespørgslerne). Bat-filen kunne man bede Windows om at eksekvere med bestemte intervaller eller på bestemte tidspunkter. Pt. vises Gekkos brugerinterface altid ved start af gekko.exe, men det ville kunne slås fra med en parameter til kaldet af gekko.exe (svarende til "silent mode").

[7] Pt. kan man kun forespørge om en given observation/periode har været ændret siden *forrige* opdatering. DST har selv en interesse i at udvide API'et mht. dette, for at reducere trækket på deres databaser.
[8] Man kunne afteste det inkrementale system i en periode, ved at køre inkremental og fuld opdatering sideløbende og sammenligne outputfilerne.

4. Menusystem og dannelse af JSON-filer

Nogle brugere vil sikkert foretrække at redigere JSON-filerne direkte, da disse ikke er specielt svære at læse. Men i mange situationer vil det sikkert være en fordel at kunne udvælge/klikke elementer til og fra via nogle menuer/tabeller som viser mulighederne, og så lade disse valg danne JSON-filerne. Gekkos menu- og tabelsystem kunne udbygges til dels at kunne danne JSON-forespørgsler via point-and-click, og menu- og tabelsystemet kunne i øvrigt også fås til at vise en slags light-udgave af Statistikbanken via gentagne kald til API'et.[9]

Et alternativ til JSON som input-format til forespørgsler kunne være et mere kompakt og Gekko-skræddersyet forespørgselsformat, hvor hver linje i en forespørgselsfil kalder en given tabel med de ønskede filtre/koder/parametre osv. Det er nok smag og behag og et spørgsmål om, hvorvidt antallet af JSON-filer er så stort, at man taber overblikket hvis de ligger i hver deres fil. I så fald kunne man måske godt definere et mere kompakt udtræksformat, hvor hver linje i dette danner en tilsvarende JSON-fil.[10]

5. Konklusion

Der er mange måder at lave interfacet mellem Gekko og Statistikbanken på, og her foreslås det, at den primære kobling foregår via forespørgsler i JSON-format og svar i PC-Axis-format. Hvis det ønskes, ville Gekko kunne generere JSON-forespørgslerne ud fra linjer i et mere simpelt filformat, eller alternativt ved at brugeren selv udvælger fra tabellerne via point-and-click (det sidste vil naturligvis tage noget mere tid at programmere). Men da JSON-formatet er ganske nemt at læse i en almindelig teksteditor, er det bestemt også en mulighed at redigere direkte i dette. Det foreslås at samle JSON-filerne i en zip-fil, primært for at holde dem samlet som en "pakke" af forespørgsler.

Gekko kaldes så med JSON-zip-filen, som udpakkes og sendes af sted som forespørgsler, hvorefter Gekko danner en ny zip-fil med en korresponderende samling af Axis-filer som den har fået retur fra Statistikbanken. Axis-formatet er valgt, fordi det er meget fleksibelt, velafprøvet og formentlig vil leve i mange år endnu (og indeholder al metainformation om data). Gekko udvides til at kunne læse Axis-filer (i hvert fald de simple af slagsen svarende til Appendiks B), og der vedtages en logik mht. tidsserienavnene, som i Gekko kan have vilkårlig længde. Gekko kan herefter gemme tidsserierne som tsdx, tsd eller andet, eventuelt efter at have omdøbt eller på anden måde transformeret tidsserierne.

Det foreslås at lægge en kopi af JSON-forespørgselsfilerne i den zip-fil, som samler Axis-outputfilerne. Derved vil en given zip-fil med Axis-filer kunne auto-opdatere sig selv (hvis man ønsker det), og der vil ikke

---

[9] Man kan forespørge på hvilke tabeller der findes, hvilke dimensioner der findes i en given tabel osv., og resultatet af dette kunne generere Gekko-menuer, som man kan klikke sig igennem. Statistikbank-API'et kan endda generere html og png som outputformat, hvilket formentlig muliggør visning af tabeller og figurer i Gekko stort set som de ser ud i på Statistikbankens hjemmeside (det skal dog lige afprøves).

[10] Eksempelvis en linje à la: "PRIS6 enhed=100 VAREGR=011200,011100 tid=*", svarende til eksemplet i Appendiks A. Sådanne linjer kan dog hurtigt blive lange, hvis der udvælges specifikke varer o.lign. fra tabellerne. En helt tredje mulighed er at bruge URL'er som svarer til udtrækket. Udtrækket i Appendiks A kunne alternativt laves vha. denne URL:
http://api.statbank.dk/v1/data/PRIS6/PX?valuePresentation=CodeAndValue&enhed=100&VAREGR=011200%2C011100&tid=*. Sådanne linjer er dog ikke særligt læselige.

være tvivl om, hvordan filen er dannet. Man kunne gøre noget lignende med den tsdx-fil, som indeholder de samme tidsserier, således at også denne indeholder information om forespørgslerne, og så den også ville kunne auto-opdatere sig selv.

For at reducere downloadtid og trækket på DST's databaser kunne man overveje inkrementale forespørgsler. Dette kan bygges på efterhånden og bør i øvrigt testes grundigt, men burde kunne laves på en ret så fejlsikker måde. Systemet vil også kunne laves, så eventuelle fejl i forespørgslerne fejler på en relativt hjælpsom måde, dvs. at Gekko videresender fejlmeddelelser fra API'et og i øvrigt også via opslag i tabelmetadata vil kunne hjælpe brugeren lidt på vej mht. hvad de lovlige parametre er til en given tabel osv.

Eksempel på JSON-fil (input)

(Filen udtrækker to speficikke prisindeks med koderne 011200 og 011100)


```
{
   "table": "pris6",
   "format": "PX",
   "valuePresentation": "Value",
   "variables": [
      {
         "code": "enhed",
         "values": [
            "100"
         ]
      },
      {
         "code": "VAREGR",
         "values": [
            "011200",
            "011100"
         ]
      },
      {
         "code": "tid",
         "values": [
            "*"
         ]
      }
   ]
}
```

Appendiks B. Eksempel på PC-Axis-fil (output)

(Dette er hvad der returneres, når når Gekko kalder api.statbank.dk med JSON-filen i Appendiks A)

```
CHARSET="ANSI";

AXIS-VERSION="2010";

CODEPAGE="Windows-1252";

LANGUAGE="da";

CREATION-DATE="20131106 09:32";

DECIMALS=1;

SHOWDECIMALS=1;

ROUNDING=1;

MATRIX="PRIS6";

AGGREGALLOWED=NO;

COPYRIGHT=YES;

SUBJECT-CODE="06";

SUBJECT-AREA="Priser og forbrug";

TITLE="Forbrugerprisindeks efter enhed, varegruppe og tid";

CONTENTS="Forbrugerprisindeks";

STUB="enhed","varegruppe";

HEADING="tid";

VALUES("enhed")="Indeks";

VALUES("varegruppe")="01.1.2 KØd","01.1.1 BrØd og kornprodukter";

VALUES("tid")="2000M01","2000M02","2000M03","2000M04","2000M05","2000M06","2000M07","2000M08","2000M
09","2000M10","2000M11","2000M12","2001M01","2001M02","2001M03","2001M04","2001M05","2001M06","2001M
07","2001M08","2001M09","2001M10","2001M11","2001M12","2002M01","2002M02","2002M03","2002M04","2002M
05","2002M06","2002M07","2002M08","2002M09","2002M10","2002M11","2002M12","2003M01","2003M02","2003M
03","2003M04","2003M05","2003M06","2003M07","2003M08","2003M09","2003M10","2003M11","2003M12","2004M
01","2004M02","2004M03","2004M04","2004M05","2004M06","2004M07","2004M08","2004M09","2004M10","2004M
11","2004M12","2005M01","2005M02","2005M03","2005M04","2005M05","2005M06","2005M07","2005M08","2005M
09","2005M10","2005M11","2005M12","2006M01","2006M02","2006M03","2006M04","2006M05","2006M06","2006M
07","2006M08","2006M09","2006M10","2006M11","2006M12","2007M01","2007M02","2007M03","2007M04","2007M
05","2007M06","2007M07","2007M08","2007M09","2007M10","2007M11","2007M12","2008M01","2008M02","2008M
03","2008M04","2008M05","2008M06","2008M07","2008M08","2008M09","2008M10","2008M11","2008M12","2009M
01","2009M02","2009M03","2009M04","2009M05","2009M06","2009M07","2009M08","2009M09","2009M10","2009M
11","2009M12","2010M01","2010M02","2010M03","2010M04","2010M05","2010M06","2010M07","2010M08","2010M
09","2010M10","2010M11","2010M12","2011M01","2011M02","2011M03","2011M04","2011M05","2011M06","2011M
07","2011M08","2011M09","2011M10","2011M11","2011M12","2012M01","2012M02","2012M03","2012M04","2012M
05","2012M06","2012M07","2012M08","2012M09","2012M10","2012M11","2012M12","2013M01","2013M02","2013M
03","2013M04","2013M05","2013M06","2013M07","2013M08","2013M09";

TIMEVAL("tid")=TLIST(M1),"200001","200002","200003","200004","200005","200006","200007","200008","20
0009","200010","200011","200012","200101","200102","200103","200104","200105","200106","200107","200
108","200109","200110","200111","200112","200201","200202","200203","200204","200205","200206","2002
```

```
07","200208","200209","200210","200211","200212","200301","200302","200303","200304","200305","20030
6","200307","200308","200309","200310","200311","200312","200401","200402","200403","200404","200405
","200406","200407","200408","200409","200410","200411","200412","200501","200502","200503","200504"
,"200505","200506","200507","200508","200509","200510","200511","200512","200601","200602","200603",
"200604","200605","200606","200607","200608","200609","200610","200611","200612","200701","200702","
200703","200704","200705","200706","200707","200708","200709","200710","200711","200712","200801","2
00802","200803","200804","200805","200806","200807","200808","200809","200810","200811","200812","20
0901","200902","200903","200904","200905","200906","200907","200908","200909","200910","200911","200
912","201001","201002","201003","201004","201005","201006","201007","201008","201009","201010","2010
11","201012","201101","201102","201103","201104","201105","201106","201107","201108","201109","20111
0","201111","201112","201201","201202","201203","201204","201205","201206","201207","201208","201209
","201210","201211","201212","201301","201302","201303","201304","201305","201306","201307","201308"
,"201309";

CODES("enhed")="100";

CODES("varegruppe")="011200","011100";

CODES("tid")="2000M01","2000M02","2000M03","2000M04","2000M05","2000M06","2000M07","2000M08","2000M0
9","2000M10","2000M11","2000M12","2001M01","2001M02","2001M03","2001M04","2001M05","2001M06","2001M0
7","2001M08","2001M09","2001M10","2001M11","2001M12","2002M01","2002M02","2002M03","2002M04","2002M0
5","2002M06","2002M07","2002M08","2002M09","2002M10","2002M11","2002M12","2003M01","2003M02","2003M0
3","2003M04","2003M05","2003M06","2003M07","2003M08","2003M09","2003M10","2003M11","2003M12","2004M0
1","2004M02","2004M03","2004M04","2004M05","2004M06","2004M07","2004M08","2004M09","2004M10","2004M1
1","2004M12","2005M01","2005M02","2005M03","2005M04","2005M05","2005M06","2005M07","2005M08","2005M0
9","2005M10","2005M11","2005M12","2006M01","2006M02","2006M03","2006M04","2006M05","2006M06","2006M0
7","2006M08","2006M09","2006M10","2006M11","2006M12","2007M01","2007M02","2007M03","2007M04","2007M0
5","2007M06","2007M07","2007M08","2007M09","2007M10","2007M11","2007M12","2008M01","2008M02","2008M0
3","2008M04","2008M05","2008M06","2008M07","2008M08","2008M09","2008M10","2008M11","2008M12","2009M0
1","2009M02","2009M03","2009M04","2009M05","2009M06","2009M07","2009M08","2009M09","2009M10","2009M1
1","2009M12","2010M01","2010M02","2010M03","2010M04","2010M05","2010M06","2010M07","2010M08","2010M0
9","2010M10","2010M11","2010M12","2011M01","2011M02","2011M03","2011M04","2011M05","2011M06","2011M0
7","2011M08","2011M09","2011M10","2011M11","2011M12","2012M01","2012M02","2012M03","2012M04","2012M0
5","2012M06","2012M07","2012M08","2012M09","2012M10","2012M11","2012M12","2013M01","2013M02","2013M0
3","2013M04","2013M05","2013M06","2013M07","2013M08","2013M09";

DOMAIN("enhed")="VPINDEKS";

DOMAIN("varegruppe")="VPPRISIND";

UNITS="Indeks";

DATABASE="Danmarks Statistik";

SOURCE="http://www.statistikbanken.dk/PRIS6";

INFOFILE="http://www.dst.dk/kvalitetsdeklaration/000898";

TABLEID="PRIS6";

DATA=

98.1 97.8 98.2 97.8 99.9 101.0 102.5 101.3 100.4 101.8 100.4 101.0 101.4 102.5 102.9 106.1 106.1
106.3 106.4 107.6 105.1 105.1 104.7 103.9 104.2 104.5 104.3 104.0 104.6 105.8 106.2 104.5 104.5
103.5 104.5 104.5 104.1 104.5 103.1 102.6 103.9 105.7 105.6 103.6 102.0 103.2 103.2 103.6 103.1
104.0 104.2 104.3 106.5 105.8 105.8 105.2 104.1 102.8 103.5 103.6 104.0 104.1 104.5 103.5 104.7
105.9 107.1 107.2 104.5 104.9 104.2 105.6 104.7 104.1 104.1 103.5 104.5 105.1 106.8 106.8 106.3
106.3 108.9 108.6 108.0 108.6 107.6 107.0 107.7 106.8 106.6 104.7 106.9 105.7 107.7 107.4 108.2
108.4 108.7 110.4 111.1 111.9 113.1 112.4 111.6 112.2 113.3 111.7 114.2 112.8 110.6 110.1 110.5
112.0 112.2 112.2 110.2 109.6 109.7 107.9 108.1 108.1 108.5 106.9 107.9 107.2 108.6 110.1 108.4
108.4 108.6 109.4 109.0 108.0 108.6 109.8 110.2 110.8 111.5 112.1 109.5 111.7 113.6 113.7 114.7
114.5 114.6 115.5 116.0 116.1 115.8 116.9 116.6 117.1 118.2 118.2 116.1 116.8 116.9 115.0 114.9
115.0 115.6 113.7 114.1
```

98.3 98.7 99.1 99.6 99.9 100.2 100.6 100.6 100.6 100.6 100.9 101.0 102.2 102.4 103.1 103.6 103.7
104.2 104.5 104.6 104.7 105.1 105.2 105.4 105.6 106.0 106.4 106.6 106.9 107.1 107.4 107.7 108.0
108.0 108.1 107.9 108.4 108.7 109.4 109.5 109.5 109.6 110.3 110.4 110.6 110.3 110.7 110.4 110.5
110.5 111.6 111.1 111.3 111.0 111.0 110.8 111.7 111.7 111.8 111.8 112.0 112.0 112.7 112.8 112.9
113.5 113.9 114.8 114.4 114.2 114.0 114.5 113.5 114.1 114.6 114.3 114.9 115.2 115.8 115.5 115.7
116.0 116.9 117.0 118.0 118.9 120.2 120.9 120.2 119.8 119.8 119.8 121.8 126.2 129.5 131.3 132.5
134.4 135.0 135.6 137.3 137.6 138.1 137.6 138.0 138.6 138.4 138.5 138.4 138.5 138.4 139.1 139.1
139.3 138.7 138.4 137.8 138.1 137.2 136.8 136.0 136.8 136.6 137.0 136.8 136.5 136.6 136.9 136.9
137.3 137.8 138.2 140.3 141.7 142.5 143.8 144.6 145.8 146.8 146.7 147.2 148.7 148.9 149.9 149.9
149.9 151.5 150.4 150.7 151.9 152.1 151.9 151.7 150.6 151.6 151.1 152.5 151.8 152.3 152.4 152.6
151.9 152.5 150.7 152.0

;